Time correlations for Hard Rod Systems

Brian Varghese

Department of Mathematics, King's College London

March 2021

Abstract

Inspired by recent results on the large-scale dynamics of interacting particles [15],[16] we study the classical hard rod gas in one dimension and compute dynamical equilibrium space-time correlations functions of conserved densities and currents. In particular, we focus on 3 instances of the hard rod gas; (1) hard rods with equal masses, (2) hard rods with alternating masses and finally (3) hard rods with random masses. The first model is completely integrable and is described by the principles of Generalized Hydrodynamics [3], while the other 2 models are non-integrable, thus, requiring a non-linear version of fluctuating hydrodynamics. In addition to equilibrium dynamics, we also study each of the models in a non-equilibrium initial state [4]. This is done by imposing a "domain wall" as an initial condition, where the system is split into two equal halves, each half being prepared in an asymptotically different state, and then joined together allowing the system to evolve for large time scales. Our project uses an iterative scheme of Monte-Carlo sampling techniques and Molecular Dynamics algorithms implemented on python to simulate the fluid flow and numerically calculate the observables of interest, as well as their space-time correlation functions.

Contents

1 Introduction

We choose the hard rod gas as a simple tractable model to study the fluctuating hydrodynamics of one-dimensional fluids. By reducing the number of dimensions, the amount of fluctuations increases giving rise to a number of interesting phenomena such as the breakdown of Fermi-liquid behaviour and the non-existence of Bose-Einstein Condensation [6], [7], [14]. The hard rod gas is a particular example of a larger class of models, namely the anharmonic chains, which are limited to interactions exclusively with adjacent particles. These one-dimensional fluids do not display any phase transitions and their static correlations exhibit rapid decay. Thus, we instead study the equilibrium space-time correlations which have non-anomalous decay. While the effects of fluctuating hydrodynamics are captured by the current-current correlations, the Green-Kubo definition for the transport coefficients does not have a closed analytic form, thus providing us with only a qualitative prediction [13]. However, over the last 20 years, there has been considerable efforts made in the fields of Monte-Carlo (MC) and Molecular Dynamics (MD) simulations helping us achieve accurate quantitative approximations.

The main goal of this paper is to use the principles of Generalized Hydrodynamics [3] and a non-linear version of fluctuating hydrodynamics [19] to numerically obtain the dynamical two-point correlation functions of conserved charges and currents. In many body systems, the dynamical correlations between observables at different space-time points are particularly important as they encode the emergent degrees of freedom and their dynamics. To obtain our results, we rely extensively on Monte-Carlo algorithms and Molecular Dynamics simulations that have been developed by me independently over the course of the project. The code is rather delicate, and while there are areas that could be improved, we see that the existing numerical results successfully match the theoretical predictions. As with any simulation, the more realistic the better, thus simulating a fluid would normally require thousands of particles averaged over many samples, however due to computational constraints, we restrict our systems to 250 particles and 1000 samples. There is a certain art to choosing the parameters for the simulation, we could in-fact reduce the number of particles and average over tens of thousands initial samples, or we could increase the number of particles and reduce the number of samples, but doing so did not yield any meaningful results, instead the parameters we chose gave the most defined structures for our correlation functions. We expect that averaging over more initial samples, say of order 10^7, give us more accurate results.

During our simulations we make the implicit assumption that there exist only 3 standard local conservation laws; conservation of mass, momenta and energy for the non-integrable models, while there are N conserved quantities for the integrable model from which we choose any 3 arbitrarily, say the same as the non-integrable model. In addition to the conserved charges we also study their

time-integrated currents and sample averaged densities at specific snapshots of time for the domain wall problem. In particular we study how the correlations change from the homogeneous initial condition to the in-homogeneous initial condition.

The study of non-equilibrium deterministic dynamics of many-body systems pose a very interesting problem and has attracted alot of interest in the past few years [4]. To investigate the non-equilibrium dynamics, we impose a "domain wall" as an initial condition. This is simply a partitioning protocol, where two systems of equal volume, with each system being prepared in asymptotically different states defined by different particle densities, are connected and allowed to evolve for large time scales. If the dynamics admits ballistic processes, then at large time scales the local observables are in a stationary state with non-zero currents - this is a non-equilibrium steady state. We investigate this further by looking at how the correlations spread over space time for the integrable and non-integrable models.

The paper is structured as follows; In section (1) we cover the basic notion of integrable and non-integrable systems, this will help us differentiate between the hydrodynamic equations for the hard rods with equal masses and the hard rods with alternating/random masses. In section (2) we cover the microscopic dynamics of the hard rods and obtain the time evolution, we also describe the specific hydrodynamic equations in Euler form and introduce the observables of interest, i.e the conserved charges and their respective currents. In addition, we introduce the "domain wall" also called the Riemann Problem and solve the dynamical equations for the integrable and non-integrable models. In section (3) we introduce the correlation functions, specifically the field-field correlations and the current-current correlations and describe how these correlations spread over space-time. In section (4) we give the details as well as the codes for the Monte-Carlo and Molecular Dynamics algorithms used to generate the results throughout the project. As an appendix, we include the calculation of the dynamical structure factor, also known as the scattering function which is obtained a Fourier transformation of the various correlation functions.

1.1 Integrable and Non-integrable Interacting systems

Hamiltonian systems can be classified into two groups, integrable and non-integrable. While a majority of models found in the physical world are non-integrable, we find that in one-spatial dimension there is a good variety of integrable models to choose from. Thus, integrability is a property of certain dynamical systems. There are several distinct definitions [9], but generally, an integrable system is characterized by a sufficient amount of conserved quantities, such that its behaviour has far fewer degrees of freedom than the dimensionality of its phase space; in simple terms, this means the system's evolution lies in a submanifold within its phase space.

Definition 1 *Let there be given a 2 dimensional real sympletic manifold (M, ω) with a Hamiltonian $H : M \times [t_i, t_f] \to \mathbb{R}$ (in principle any even dimensional manifold work). The time evolution of this system is governed by the Hamiltonian's or equivalently Louiville's equations of motion. We can now define the notion of complete integrability aka Louiville integrability as the existence of n independent real functions I_i for $i \in \{1, 2, 3..., n\}$ such that*

$$\{I_i, I_j\} = 0 \quad \text{for} \quad i, j \in \{1, 2, 3..n\} \tag{1.1}$$

where

$$\{I_i, I_j\} = \sum_{i,j=1}^{n} \left(\frac{\partial p_i}{\partial q_s}\frac{\partial p_j}{\partial p_s} - \frac{\partial p_i}{\partial p_s}\frac{\partial p_j}{\partial q_s}\right)$$

These I_i 's which we call action angle variables, are a special set of canonical coordinated on the phase space of an integrable system, where levels of invariant foliation are tori, such that the invariant tori are the union of level sets of the action angle variables.

Definition 2 *Given a fixed point $x_0 \in M$, under mild regularity assumptions, there always exists locally (In a sufficiently small Darboux neighbourhood of x_0) an n-parameter complete solution to the Hamiltonian's principle function*

$$S(q^1, q^2..., q^n; I_1, I_2, ..., I_n; t) \tag{1.2}$$

to the Hamiltonian-Jacobian equation, where $I_i, i \in \{1, 2, ..n\}$ are integration constant. This leads to a local version of (1.1). The main point being that (9) being a global property is very rare, while (1.2) being a local property is more generic.

There are 3 main features that characterize Integrable hamiltonian systems

1. The existence of a maximal set of conserved quantities (the usual defining property of complete integrability).
2. The existence of algebraic invariants, having a basis in algebraic geometry.

3. The explicit determination of a solution in an explicit functional form, often referred to as solvability.

Integrability is a stronger condition than the notion of solvability for an initial value problem. The very presence of integrability implies the absence of chaotic orbits. More precisely, all bounded orbits are quasi-periodic i.e. lying on an invariant torus. The idea of chaotic systems, which define most generic dynamical systems observed in the physical world, generally admit no conserved quantities and are asymptotically intractable. This is because an arbitrary small perturbation in initial conditions may lead to arbitrarily large deviations in their trajectories over sufficiently large time scales. Thus, complete integrability is a non-generic property of dynamical systems.

In physics, particularly statistical mechanics, there are two methods used to solve completely integrable Hamiltonian systems; The classical thermodynamic Bethe-Ansatz approach, in its modern sense based on the Yang-Baxter equations and the other method being the quantum inverse scattering method.

For the purpose of our project, we consider the classical example of the hard rod fluid. For N hard rods with equal masses, the system admits N conservation laws labeled by its velocities. By definition such a system is integrable and has already been solved using classical Thermodynamics Bethe-Ansatz. We find that we can remove the integrability by introducing a sufficient amount of chaos in the system, this is achieved by adjusting the mass of each particle. The first non-integrable case is a mixture of hard rods with alternating masses, it is known that a mass ratio greater than 3 is sufficient for destroying the integrability of the hamiltonian. In our simulations we take a mass ratio of 5 i.e. $m_1 = 1$ and $m_2 = 5$. The other non-integrable model is a mixture of hard rods with random masses. This makes the dynamics extremely chaotic. As a consequence of non-integrability, there are a finite number of conservation laws, allowing us to modify (1.1) to give

$$I_i, \quad \text{for} \quad i \in \{1, 2, 3\} \tag{1.3}$$

The i's are the conserved fields; mass, momentum and energy. We make the central assumption that there are no more than 3 conservation laws, however this is very difficult to show.

As a special scenario, specific to the equal mass hard rods, we include a brief discussion of interacting and non-interacting integrable systems. The interacting equal mass hard rods have previously been mapped to a non-interacting gas, for which the correlations have been extracted in. The underlying difference between interacting and non-interacting systems is that the Onsager matrix is zero for the latter and non-zero for the former [18].

2 Hard rod fluid

We begin this section by defining the classical hard rod model for a one dimensional fluid, the hard rods have length "a" > 0 , positions $x_i \in \mathbb{R}^+$ and velocities $v_i \in \mathbb{R}$. They move freely except for elastic collisions, when two hard rods collide, the exchange velocity upon impact, conserving momentum and energy. The hard rod model has a certain level of convenience when it comes to simulating, this is because we can label each of the hard rods with an index and it remains in the same order throughout time evolution. This is because no two hard rods can overlap let alone pass through one another. Thus, the system remains labelled as $0 < x_0 < x_1 < \ldots < x_n < L$. The i^{th} particle moves along a straight line $\dot{x_i} = v_i$, interrupted by frequent jumps back and forth. For a system of N hard rods, this implies that there are N conserved quantities labelled by their velocities.

To be more precise, the above case was specific to hard rods with equal masses. When the rods have random or alternating masses, the particles move along a straight line $\dot{x_i} = m_i v_i$, whereby two hard rods collide with each other results in the exchange momentum instead of velocity, thus momenta are conserved rather than the velocity. We can get back the integrable case if we were to set m=1. It is interesting to note that the hard rods with equal mass which is an interacting system can be mapped to a non-interacting gas in the ideal gas limit – 0.

The hard rods interact with the following inter-molecular pair potential [8]

$$V(x - x') = \begin{cases} \infty, \text{ if } |x - x'| < a \\ 0, \text{ if } |x - x'| \geq a \end{cases} \tag{2.1}$$

Since we simulated an ensemble with a fixed volume, we can convert the problem into an infinite potential well, where the potential energy of the particle becomes infinite if is found outside the ensemble giving it a PDF of 0. This means the system now follows

$$V(x) = \begin{cases} \infty, \text{ if } x \leq a \\ 0, \text{ if } \frac{a}{2} \leq x \leq L - \frac{a}{2} \\ \infty, \text{ if } x > L - \frac{a}{2} \end{cases} \tag{2.2}$$

For our simulations, we come up with two different ways to generate initial configurations of hard rods. The first method which is more accurate but has a much higher code complexity generates highly random positions of a fixed number of particles for every sample. However, this could not be used to simulate a large number of particles instead we use an idea to place a particle at an initial position and then add a random positive distance to ensure no overlap, the

major drawback with this method is that the number of particles keep fluctuating, however this is not expected to have a huge impact on the actual physics, since the density remains "almost" the same for every sample. We also assign velocities to each of the particles sampled from a Maxwell Boltzmann distribution which is just a gaussian function with mean = 0 and standard deviation = $\sqrt{(k_B \times T)/m}$, where k_B is the Boltzmann constant, T is the temperature and m is the mass of the particle. The velocity distribution itself changes for every sample.

Thus, we specify an equilibrium state when the positions and velocities have been assigned to each particle. Under this equilibrium measure, velocities are independent of positions and the velocities themselves are independently distributed. The positions on the other hand are correlated because of hard core repulsion. The infinite volume system is obtained by extending N, L $\to \infty$ with the density remaining constant. We average over many initial realizations for the dynamical observables as well average over time steps for specific observables for better visualization.

The canonical ensemble [4] is given by the probability measure

$$Z^{-1}\chi((q_i - q_j) \geq a, 1 \leq i < j \leq N) \prod_{j=1}^{N} h(p_j) \frac{1}{N!} dq_j dp_j \tag{2.3}$$

where Z is the normalization factor (Canonical Partition Function), χ is the indicator function of the particular set, $h(p_{i+1})$ is the single particle velocity distribution. $\chi = e^{\frac{-U}{k_B T}}$ is 0 if any two hard rods overlap and is exactly 1 otherwise. Without loss of generality we can integrate over space and multiply by N!. x_j must lie to the right of $x_j - 1$ and to the left of $Y_j = L - 1/2a - (N-j)a$

We can visualize the hard rods lying in space using Monte-Carlo sampling techniques.

We can derive various thermodynamic properties for the system specifically the partition function, the free energy and the equation of state as follows.

The partition function

$$Z(T, L, N) = \frac{\lambda_T^N}{N!} \int_0^L dx_1 ... \int_0^L dx_N \chi(x_1, x_2, ...x_N) \tag{2.4}$$

We can solve this for N hard rods of equal length a as

$$
\begin{aligned}
Z(T,L,N) &= \lambda_T^N \int_{a/2}^{Y_1} dx_1 \int_{x_1+a}^{Y_2} dx_2 ..... \int_{X_{N-1}+a}^{Y_N} dx_N \\
&= \lambda_T^N \int_{a/2}^{Y_1} dx_1 \int_{x_1+a}^{Y_2} dx_2 ..... \int_{X_{N-1}+a}^{Y_N} dx_N \\
&= \lambda_T^N \int_{a/2}^{Y_1} dx_1 \int_{x_1+a}^{Y_2} dx_2 ... \int_{X_{N-1}+a}^{Y_{N-1}} dx_{N-1}(Y_{N-1} - x_{N-1}) \\
&= \lambda_T^N \int_{a/2}^{Y_1} dx_1 \int_{x_1+a}^{Y_2} dx_2 ..... \int_{x_{N-3}+a}^{Y_{N-2}} dx_{N-2} \frac{(Y_{N-2} - x_{N-2})^2}{2} \\
&= .... = \frac{\lambda_T^N}{N!}(X_1 - \frac{a}{2})^N = \frac{\lambda_T^N}{N!}(L - Na)^N
\end{aligned}
\tag{2.5}
$$

Where the λ_T^N is the momenta integrated over the subspace $x_1 < x_2 < .... < x_N$. Thus we get $\lambda_T = \sqrt{\frac{h^2}{2\pi m k_b T}}$

We can compute the free energy as

$$
F = -k_B T \log_e(Z) = -N k_B T(\log_e \lambda_T + 1 + \log_e(L/N - a) \tag{2.6}
$$

The equations of state

$$
P = -(\frac{\partial A}{\partial L})_{N,T} = \frac{N k_B T}{L - Na} \tag{2.7}
$$

where A is the Helmholtz energy function, L is the length of the system and a is the size of each particle. Once we have the pressure of the system P, we can calculate the compressibility factor

$$
\beta = \frac{PL}{N k_B T} = \frac{1}{1 - \eta} \tag{2.8}
$$

where $\eta = \frac{N \times a}{L}$ is the fraction of the total volume occupied by the rods.

Under this equilibrium measure, velocities are independent of positions and the velocities themselves are independently distributed. The positions on the other hand are correlated because of hard core repulsion. The infinite volume system is obtained by extending $N, L \to \infty$ with the density remaining constant. We average over many initial realizations for the dynamical observables as well average over time steps for specific observables for better visualization.

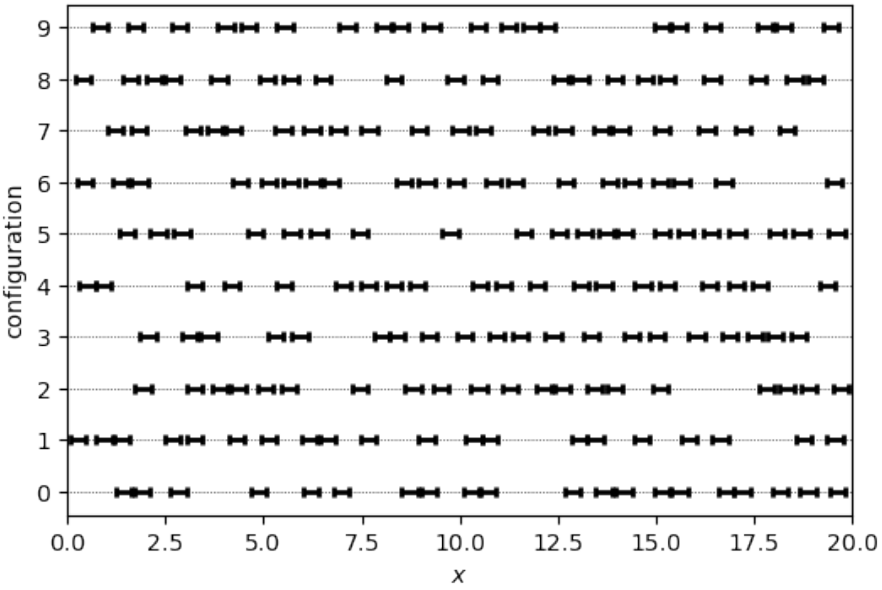

Figure 1: These are various configurations of hard rods. We take 10 different Monte-Carlo iterations, with each rod having a length of 0.2 in a volume of 20. For this particular Monte-Carlo algorithm we able to fix the number of particles, but for simulations with larger particles we keep the number of particles fluctuating around the mean density. This was done to make the code more efficient allowing us to run more than 1000 simulations with time evolution in a couple of hours

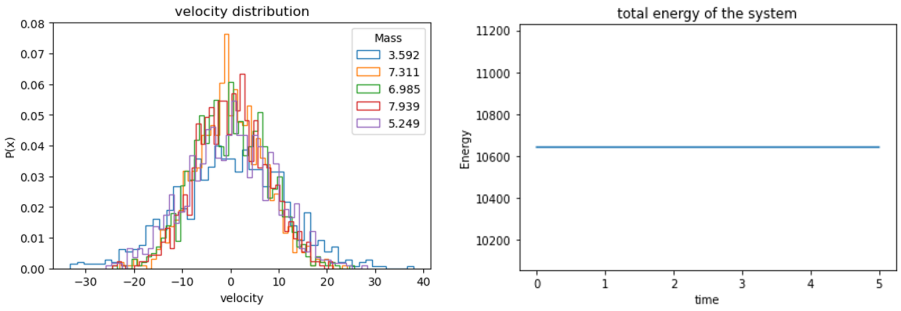

Figure 2: We can also visualize the velocity distribution, where we have generated a 1000 Monte-Carlo samples for 5 particles of random masses. We note that the velocity distribution keeps fluctuating for each Monte-Carlo realization of initial states. The second figure is the total energy over time, which remains constant as it is simply the sum of kinetic energy at every time step

2.1 Microscopic dynamics

The dynamics of the system can be obtained as a Hamiltonian of the form

$$H = \sum_{i=1}^{N} \frac{p_i^2}{2m} + \sum_{i=2}^{N} V(x_i - x_{i-1}) \tag{2.9}$$

where p_i is the momentum of the particle i and V is the potential energy between two particles in position x_i and x_{i-1}

If we were to use quantum Monte-Carlo simulations which are explained in the conclusion, we would instead use the Schrodinger's equation for the hard rod gas as

$$H = -\frac{\hbar}{2m} \sum_{i=1}^{N} \frac{\partial^2}{\partial r_i^2} + \sum_{i=2}^{N} V(x_i - x_{i-1}) \tag{2.10}$$

where $\hbar$ is the reduced Planck's constant.

As mentioned previously the defining feature of the hard rods is that particles can neither overlap nor pass through one another, as a result the labeling of particle positions remain ordered from left to right throughout the time evolution of the system. This means that collisions can only occur between neighboring particles i and i+1. We therefore only need to consider masses and velocities of the adjacent particles. An algorithm for the masses and the velocities of the particles after the jth collision can be expressed as

$$\mu_{i,i+1} = \frac{m_i m_{i+1}}{m_i + m_{i+1}}, \quad \text{and} \quad v_{i,i+1}^j = v_i^j - v_{i+1}^j \tag{2.11}$$

after the jth collision particle i is approaching particle i+1 if and only if the $v_{i,i+1}^j > 0$. This is a necessary condition for the i^{th} and $(i+1)^{th}$ particles to collide but not a sufficient condition since another collision may occur before i^{th} and $(i+1)^{th}$ particles collide.

Using the laws of conservation of momentum and energy we can derive the velocities of the i^{th} and $(i+1)^{th}$ particles after the j^{th} collision as follows

$$v_i^{j+1} = v_i^j - 2\frac{\mu_{i,i+1} v_{i,i+1}^j}{m_i}$$
$$v_{i+1}^{j+1} = v_{i+1}^j + 2\frac{\mu_{i,i+1} v_{i,i+1}^j}{m_{i+1}}$$

Focusing on the Eulerian scale i.e. large-distance, long-time scales which emerge when fluctuating fields change in space. The scale is controlled by the length of the particle "a" in the following manner; it emerges, in some large region, when the space between rods throughout this region, in mean with respect to the local generalized equilibrium, times the mean variation length of the rod density, is much greater than a^2. We calculate the local equilibrium in

2 ways: averaging over time, and space-time "fluid cells", large enough to give negligible fluctuations, but small enough so that local states are slowly varying; and additionally we also average over realizations of the dynamics with different initial conditions determined by a given distribution, thus allowing the size of fluid cells to be reduced arbitrarily.

To get a better understanding of how the hard rods interact with one another we can solve the Hamiltonian essentially evolving the system over time. For this particular simulation we generate an initial configuration of 16 particles with velocities chosen from a Maxwell Boltzmann distribution and allow the system to evolve for time t=10. The resulting solution is a set of trajectories formed when particles collide with one another or the boundary of the system. This also acts as an additional check to ensure our system is interacting with its surrounding properly. To be more precise when we say that a system is evolving for time t=10, we mean the number of time steps. For this example, we take the size of each time step, dt=0.001 while the total time is 10. Thus, the number of time steps is total time / step size = 10,000. We can increase the number of steps by reducing the size of each step to make our simulations more accurate however this also means the information stored and the time taken to compute increases.

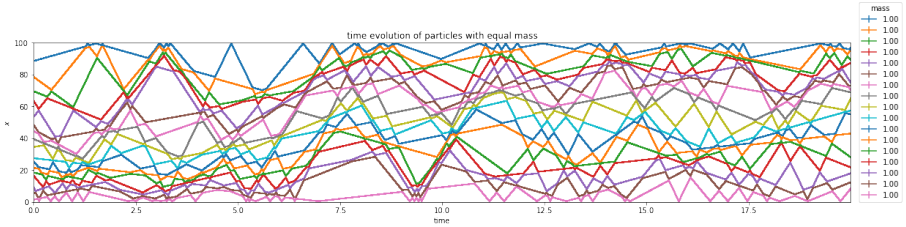

Figure 3: Time evolution of hard rod gas with equal masses

It was shown by Jepson [8] that the equal mass hard rods were a special case where the interacting system could be mapped to a gas of non-interacting particles. This allows us to compute exact results for the velocity correlation functions. The mapping to a non-interacting gas can be explained as follows; a particle at position x with a velocity v travels to x' at time t, such that $x' = x + vt = x + \frac{\sigma_t}{v}$. Quite recently the behaviour of correlations for density, momentum, stretch and energy have been studied for the toda chain and the equal mass hard rods.

As a result of this behaviour that allows us to map to a non-interacting gas, the trajectories for the equal mass look quite interesting. We can think of it as each particle following a straight-line trajectory barring collision with the boundary. It is seen in a close up of Fig 3, where the black line signifies the trajectory of a non-interacting particle, while the different colored lines are trajectories of different particles. These "straight lines" are unique to Integrable systems as we will soon find out.

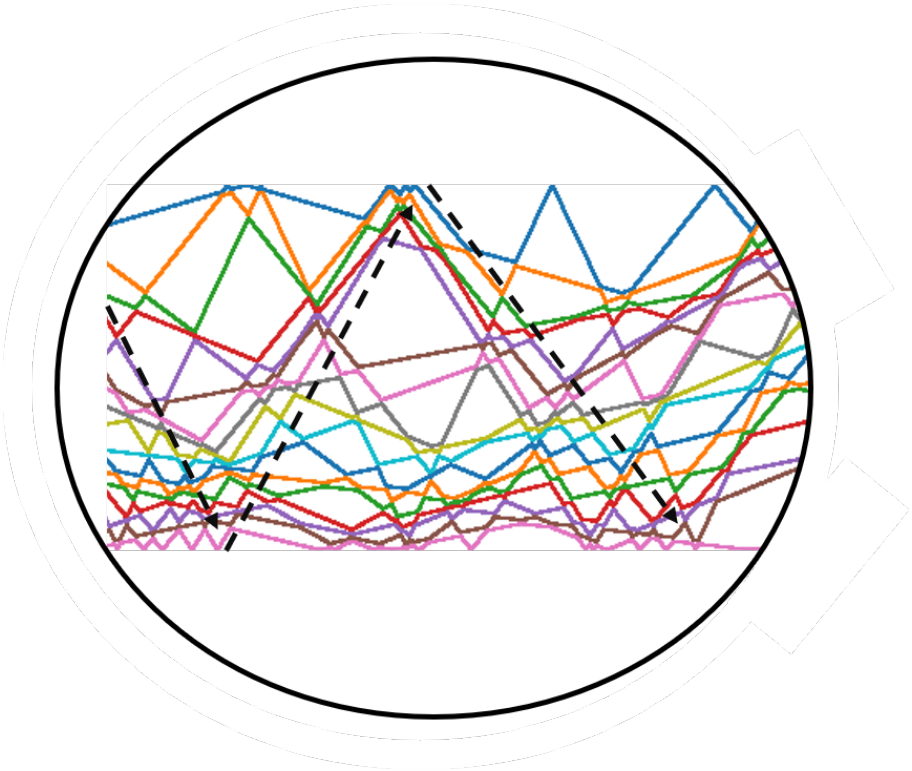

Figure 4: Tagged particle dynamics

On the other hand, these are the particle trajectories for a mixture of 18 hard rods, where each hard rod has a mass drawn from a random normal distribution shifted to the right so that all the values sampled are positive, further we scale the distribution by a factor of 10, essentially giving us random masses from (0,10]. This means that the system is no longer integrable and instead admits chaotic orbits. This is clearly seen in the figure; we cannot map this system to a non-interacting gas i.e. there is no trajectory pattern that we can follow as one single non-interacting particle. Each of the trajectories are "curved" rather than the straight pattern we saw earlier. It would be interesting to see whether these trajectories could be calculated explicitly but given that it would be a many-body problem too complex to solve analytically, we are content with numerical simulations. Something else that we can note is the collision rate, which is given by solving the collision integral. In our simulations for the equal mass hard rods we observed 788 collisions compared to the 303 collisions for the hard rods with random masses. Looking at the graph we expect the tagged particle diffusion, which is basically a measure of how much a particle gets displaced after a certain amount of time is more for the hard rods with random masses.

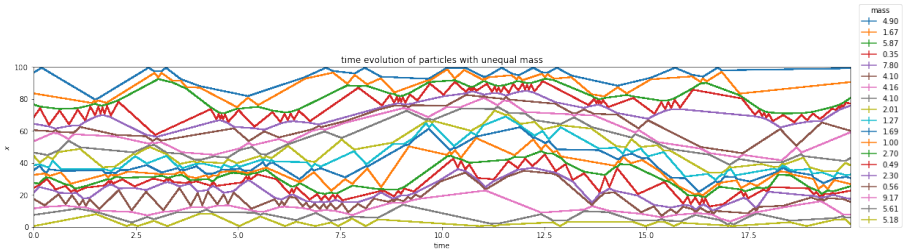

Figure 5: Time evolution for a system of 18 hard rods with random masses.

2.2 Hydrodynamics

Hydrodynamics studies fluids in motion such that the fluid velocity at each point in space does not change with respect to time. While a rigorous mathematical proof of its validity is yet to be discovered, the principles of hydrodynamics are well established and forms the backbone of fluid dynamics. It has found a wide range of applications in physics including high-energy, statistical and condensed matter physics as well as important engineering applications ranging from predicting weather patterns to understanding nebulae in interstellar space and modelling fission weapon detonation. Before the 20th century hydrodynamics was synonymous with fluid dynamics because many of the principles in hydrodynamics can also be applied to gases. The foundational axioms that govern fluid flow are the conservation laws, specifically conservation of mass, linear momentum and energy. These are based on classical mechanics and are expressed using Reynolds transport theorem.

The Euler equation in one dimension for the mass conservation takes the form of a continuity equation [5]

$$\frac{\partial \rho(x,t)}{\partial t} + \frac{\partial [v(x,t)\rho(x,t)]}{\partial x} = 0 \tag{2.12}$$

Where $\rho(x,t)$ is the density of the fluid at a particular point in space and time and $v(x,t)$ is the velocity field of the fluid. We can simplify this equation by replacing $j(x,t) = \rho(x,t)v(x,t)$ in (1) to get

$$\frac{\partial \rho(x,t)}{\partial t} + \frac{\partial j(x,t)}{\partial x} = 0 \tag{2.13}$$

For fluids that are sufficiently dense and have flow velocities small in comparison to the speed of light, the momentum equations for Newtonian fluids are described by the Navier-Stokes equations [11]; a set of non linear differential equations that describe the flow of a fluid whose stress depends linearly on the flow velocity gradients and pressure. The Navier-Stokes equation is a famous millennium problem in mathematics, and finding a closed form solution to it will land you a $1,000,000. While an analytic solution does not exist, it works

surprisingly well to model fluid flow and is used extensively in computational fluid dynamics. The Navier Stokes equation is expressed as

$$\frac{\partial v}{\partial t} + v\frac{\partial v}{\partial x} + \frac{1}{\rho}\frac{\partial P}{\partial x} - v\frac{\partial^2}{\partial x^2}v = 0 \tag{2.14}$$

Where we have dropped the space time dependency for v and ρ. P is the pressure term that depends on the density ρ

Finally we have the continuity equation for the conservation of energy,

$$\frac{\partial E}{\partial t} + \frac{\partial(Ev + Pv)}{\partial x} = 0 \tag{2.15}$$

Where E is the total energy per unit volume i.e sum of internal and kinetic energy.

The Euler equations for hydrodynamics are based on Newton's second law, however this gives rise to the "The paradox of reversibility". Since $v(x,t)$ is a solution we expect the reverse of the flow $-v(x,-t)$ to also be a solution to the Euler equations, however this is not the case. This is because in addition to Newtons equations there is also an assumption of thermalization. When we look in local region say a fluid cell containing some amount of molecules, this state depends only on some overall velocity and temperature, the information about the initial conditions is completely lost. This loss of information is completely irreversible and is described as the diffusive term. The main idea is that when we try to describe the large scale dynamics using Newton's equations the information cascades into smaller scales, where the information is lost, thus solving this open problem related to finding the scale at which this information is still present. There is no proof of this result and it remains an open problem.

2.2.1 Hydrodynamic equations for the hard rods

The specific hydrodynamics for the hard rod model was derived in 1982 by Boldrighini, Dobrushin and Suhov [1]. Under certain conditions on the initial states P_ϵ (probability measures on the phase space of the infinite hard-rod system, or, in the probabilistic terminology, of random marked point processes with marks from $\mathbb{R}$) where $\epsilon \to 0$ is an appropriate scaling factor, the limit of

$$f_t(x,v) = \lim_{\epsilon \to 0} \rho T_{\epsilon^{-1}t} P_\epsilon(\epsilon^{-1}x, v) \tag{2.16}$$

exists and gives us the unique solution of the differential equation

$$\frac{\partial f_t(x,v)}{\partial t} = (A f_t(x,v))(x,v) \tag{2.17}$$

$\rho T_{\epsilon^{-1}t} P_\epsilon(\epsilon^{-1}x, v)$ is the moment function giving us the density of the particles at specific point in space and at micro time $T_{\epsilon^{-1}t}$. A is a non-linear first order differential operator such that

$$(Af_t(x,v))(x,v) = -\frac{\partial}{\partial x}[f_t(x,v)(v + a\int_{\mathbb{R}} dw(v-w)f_t(x,v)) \\ \times (1 - a\int_{\mathbb{R}} dw' f_t(x,w'))^{-1})] \quad (2.18)$$

this is interpreted as Euler's equations for the hard rod gas. We can think of the Navier Stokes equation as a correction to the Euler equation (1.6)

$$\frac{\partial f_t^{(\epsilon)}}{\partial t} = Af_t^{(\epsilon)} + \epsilon B f_t^{(\epsilon)} \quad (2.19)$$

where B is a non linear second order differential equation

$$Bf_0(x,v) = \frac{a^2}{2}\frac{\partial}{\partial x}[\int dw|v-w|f_0(x,v)(1-\rho' a)^{-1} \\ \times (\frac{\partial}{\partial x}f_0(x,v) - f_0(x,v)\frac{\partial}{\partial x}\int dw|v-w|f_0(x,w)(1-\rho' a)^{-1})] \quad (2.20)$$

We can also calculate 4 important matrices that contain all the thermodynamic properties of the system. The first is the static co-variance matrix C.

$$C_{i,j} = \sum_x < q_i(x,t)q_j(L/2,0) > \quad (2.21)$$

where $< .,. >$ denotes the sample average.

Apart from the static co-variance matrix we can also define the field-current co-variance matrix by integrating the field-current correlation in space.

$$B_{i,j} = \sum_x < q_i(x,t)J_j(L/2,0) > \quad (2.22)$$

Now we have the tools to calculate the flux jacobian A and the Drude Weight D

$$A_{i,j} = BC^{-1} \quad \text{and} \quad D_{i,j} = A^2 C \quad (2.23)$$

The A matrix is particularly important as the eigenvalues which are positive are exactly equal to the velocity of the propagators in the correlation functions. Numerically calculating such a matrix is non-trivial and we tried to verify its results, however it required a very precise definition and a much more specific algorithm.

2.3 Generalized Hydrodynamics

The principles of conventional hydrodynamics is limited in the sense that it can only be applied to models that display a finite amount of conservation laws. To study integrable models, we need a theory that applies the standard hydrodynamic principles to systems that admit an infinite amount of conservation laws, and Generalized Hydrodynamics does just that.

In this section we give a brief overview of generalized hydrodynamics [GHD] which was developed by Castro-Alvaredo, Doyon and Yoshimura [2] has already been successfully applied to the hard rods with equal masses, both in equilibrium and non-equilibrium initial states. In addition to the hard rods it has also been applied to numerous other models like the toda chain and the Leib-Luttiger models. The theory, which is valid at the Euler-scale, helps us describe transport coefficients for integrable systems, which due to an infinite amount of conservation laws admit dynamics very different from that of conventional hydrodynamics. The theory uses the Thermodynamics Bethe-Ansatz (TBA) approach to compute the root density which in turn encodes the full thermodynamics of the system.

The TBA describes the stationary states of the integrable system as generalized Gibbs ensembles. Of particular physical interest are the space-time correlations of the conserved quantities when in thermal equilibrium. More recently, Molecular dynamics for a 1000 particles have been performed in [10]. In principle, the molecular dynamics can be computed by solving newton's equations of motion however it also requires roughly 10^7 samples for initial states to average over. Unfortunately, we did not have the computational power to do this, however we included code that would help us store simulation data recursively allowing us to compute as many Monte-Carlo samples as we require at the expense of the time taken to finish the entire simulation. In the end, we were not able to implement and test it. As a result, the energy-energy correlations have a lot of noise and their results become inconclusive for a few sets of models.

The basic hydrodynamic assumption is that if we take any particular region in Fig[3], and calculate some local observable, this region would be in a state where entropy is maximized. These states are non-trivial and depend on the model however they are controlled by the conserved quantities of the system.

This relation between maximal entropy states and conserved quantities is expressed in Gibbs form where

$$\rho = e^{-\sum_i \beta^i(x,t) Q_i} \tag{2.24}$$

Where Q_i are the conserved quantities and $\beta_i(x,t)$ are Lagrange parameters. These are explained in more detail in the later sections.

We make a slight change in our simulation parameters when calculating the correlation functions. Instead of calculating the charge and currents at a specific point in space, we instead use a 'fluid cell'. Let N(x,t) be a mesoscopic (smaller than macroscopic but larger than microscopic) fluid cell where we expect to find

on average the $\rho(x,t)$ for every fluid cell. The fundamental equations of GHD are obtained as follows, we make the assumption that when the variation scales of the state become large enough in space-time, then we may assume that on every "fluid cell" – large enough to contain a thermodynamic amount of microscopic particles, but small enough so that their state vary slowly in space-time – the state has maximised entropy with respect to all available linearly extensive conserved quantities. This is expressed in the approximation, where averages of observables at (x,t) are evaluated by calculating their average in a maximal entropy state whose Lagrange parameters depend on (x,t). In our system this means that our root density $\rho_p(P) \to \rho_p(p,x,t)$ and $n(p) \to n(p,x,t)$. The quasiparticle density, per unit of length and momentum, now depends not just on the momentum, but on space-time as well. This function of p tells us in what state the fluid cell at space-time point (x,t) is. Once we have this we can extended Euler hydrodynamics equations by replacing q and j with q_i and j_i. Bringing the space derivatives inside the momentum integral we get

$$\int dp h_i(p) \frac{\partial \rho(p,x,t)}{\partial t} = \int \frac{\partial [v^{eff}(p,x,t)\rho_p(p,x,t)]}{\partial x} \tag{2.25}$$

Using completeness of the set h_i we then get the GHD equations as

$$\frac{\partial \rho_p(p,x,t)}{\partial t} = \frac{\partial [v^{eff}(p,x,t)\rho_p(p,x,t)]}{\partial x} \tag{2.26}$$

Where $\rho_p(p,x,t)$ represents the density of the fluid at a specific point in space time. v(p,x,t) is the effective velocity evaluated in the state at (x,t). The equation (2.26) is highly non-linear as v^{eff} depends non-linearly on $\rho_p(p,x,t)$ however the equation is homogeneous in space-time. Solving this equation means setting an initial condition on time t being 0.

2.4 Conserved fields; densities and currents

We now move on to the observables of interest i.e. the conserved fields and currents. Conservation laws are fundamental to physics, where they describe phenomena which can and cannot occur. For example, the conservation law of energy states that the energy of a system isolated from its surroundings cannot change. A particularly important result is concerned with Noether's theorem, which states that there is a one-one bijection between each conservation law and differentiable symmetry of nature. For example, the conservation of energy arises from the time-invariance of physical systems, while conservation of angular momentum implies rotational invariance.

We define the locally conserved charges and their corresponding currents. A charge is any type of quantity that corresponds to time independent generators of a symmetry group which commute with the Hamiltonian. Charges are often denoted by Q and are indexed by i for each conserved quantity. For an integrable system I $\in \{1,2,3..N\}$ where $N \to \infty$. While for non-integrable systems N is

finite. As a consequence of Noether's theorem, if a system has a symmetry of some sort say Q then it can be mapped to a conserved current. The amount of something that flows in a current is the "charge".

We become particularly interested in the charge density, which is the amount of electric charge per unit length. Since our project exclusively lies in one-spatial dimension, we focus on the linear charge density, this quantity varies with time and position. Due to the conservation of electric charge, the charge density in any volume can only change if an electric current flows into or out of the volume. It is this phenomenon that is expressed via the continuity equation (1.1).

The charge density is the ratio of an infinitesimal electric charge to an infinitesimal line element $\frac{\partial Q}{\partial L}$. Integrating this quantity over space gives the total charge Q of the space. Of course, the key step to adapting this definition to numerical methods is by discretizing this expression. Let $\Delta = |x_n - x_m|$ be the interval between any 2 arbitrary particles, we define the number density as $n(x,t) = \sum_{i=0}^{N} \delta(x - x_i(t))$. This tells us how many particles are present in Δ at time t. Similarly, we can obtain the momentum fields and energy fields by multiplying the number density with the particle's momentum and energy respectively.

Thus, the conserved fields can be expressed as [17]:

$$\begin{aligned}\rho_0(x,t) &= \textstyle\sum_{i=0}^{N} \delta(x - x_i(t)),\\ \rho_1(x,t) &= \textstyle\sum_{i=0}^{N} \delta(x - x_i(t))p_i,\\ \rho_2(x,t) &= \textstyle\sum_{i=0}^{N} \delta(x - x_i(t))E_i\end{aligned}$$

We can extract $Q_i = \int(\rho(x,t)dx)$

Here we employ the use of large deviation theory to study fluctuations of the conserved current, J(x, t). Standard examples of J(x, t) would be mass, momenta and energy in equilibrium thermodynamics. According to large-deviation theory, these quantities have a PDF that are exponentially peaked at an almost sure value.

A system tends to reach thermal equilibrium when the entropy is maximized with respect to all local conservation laws, which are in turn characterized by Lagrange parameters β_i these states are of Gibbs form $e^{\sum(B^i Q_i)}$

The currents of the system is obtained by multiplying the density fields with the velocity fields. Theoretically $j(x,t)$ is the number of particles which cross a particular point x from left to right subtracted from the number of particles which cross the same point from right to left. These are precisely the current fields. (The current field J, also known as the flux of q can also be thought off as the amount of q crossing a unit area in unit time)

$$j_0 = \sum_{i=0}^{N} \delta(x - x_i(t))v_i, j_1 = \sum_{i=0}^{N} \delta(x - x_i(t))p_i v_i, j_2 = \sum_{i=0}^{N} \delta(x - x_i(t))E_i v_i \quad (2.27)$$

In statistical mechanics, these conservation laws are related given by the continuity equation $\frac{\partial \rho_i}{\partial t} + \frac{\partial j_i}{\partial x} = 0$ for $i = 0, 1, 2$

We can visualize this for our hard rods in 2 different ways. First, we take the time integrated fields and currents averaged over different initial conditions. Then we calculate the average observables for specific times. We can see that the simulation data are more or less expected from what we know in theory.

The time integrated charges and currents are calculated as follows

$$\rho_i(x) = \frac{1}{T} \sum_{t \in T} \rho_i(x, t) \quad \text{for} \quad i \in \{1, 2, 3\} \tag{2.28}$$

Now for the actual simulation data. When a system begins in a homogeneous state it remains in a homogeneous state throughout the time evolution, thus the "time-averaged' densities remain constant at 0.5. We also note that the time-integrated energy density for each model remains constant over space. We also observe that the momentum keeps fluctuating, in principle we can keep the momentum constant by normalizing the set of velocities generated during each Monte-Carlo iteration. The momentum is 0 at the ends of the volume (because?). For the currents on the other hand, the densities and energy are fluctuating. While the moving average looks more or less constant, we could attribute these fluctuations (noise) due to the low number of initial samples we have averaged over. It actually requires us to run an order of 10^7 simulations to actually get conclusive results for the energy of the system. This also affects the energy-energy correlations in the later part of the project.

We note that for the following set of simulations we obtained an average of 50 particles in a volume of 100 sampled over 500 different initial conditions. This means that figure models the equations. While it is not expected to make any drastic changes to our simulations, we note that the number of particles is not fixed for every realization of an initial configuration, rather the number of particles remain close to the density we choose. This was because we could not come up with an efficient Monte-Carlo algorithm that could choose a fixed number of particles with random positions. We also note that the positions themselves are actually natural numbers, however we were able to fix this problem by allowing the system to thermalize and measuring the results after a few time steps.

$$< \rho_i(x) >= \frac{1}{N} \sum_{j=1}^{N} \rho_i^j(x) \quad \text{for} \quad i \in \{1, 2, 3\} \tag{2.29}$$

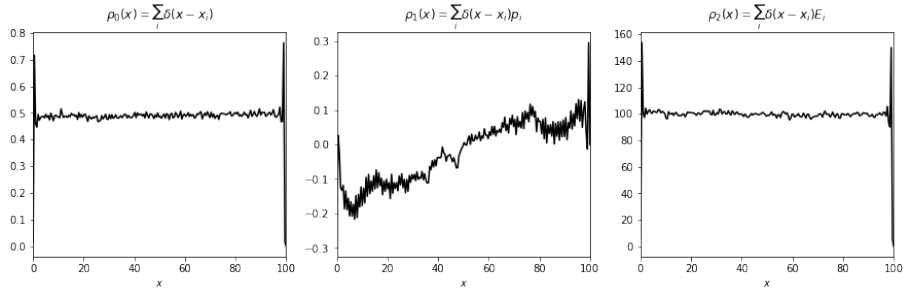

Figure 6: These are the time integrated charges for density, momenta and energy for a system of hard rods with equal masses. The charges for the other models look extremely similar to the equal mass case, thus we only include the results for equal mass hard rods. The density remains constant throughout time evolution across all the initial states. The momenta keeps fluctuating however we can note that the momenta starts and ends at 0. The momenta also increases as we move from the high density region to the low density region. This may mean that the the fluctuations of momenta are lower in the high density region compared to the low density region. The time integrated energy charge remains constant.

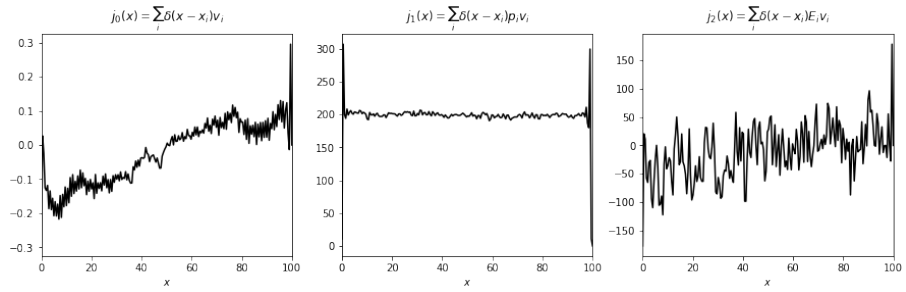

Figure 7: The time integrated currents for the density, momenta and energy are actually similar to the charges and this is by design. Since we multiply the charge by the velocity to get the current. We expect the density current to look like the momenta charge, the momenta current to look like the energy charge. The energy current looks to have a constant moving average with some white noise. This is probably due to a low sample average.

2.4.1 Riemann Problem

So far, we have focused on systems that have equilibrium initial states, thus it is natural to study the system far from equilibrium. Many body physics far from equilibrium poses some of the most interesting and challenging questions in modern science. To investigate this problem, we move on to a more interesting initial state. A natural way to do this is by using a portioning protocol, where we split the system into two equal halves, each half being prepared in an asymptotically different state. Essentially this means that the densities of the two halves are different from one another. We then join both the halves and allow the system to evolve for large time scales. What is particularly interesting is the emergence of shock waves and rarefaction patterns, which is absent in systems that have equilibrium initial states. In one-dimension integrability strongly affects non-equilibrium physics as explained via the newton cradle experiment on cold atomic gasses. Relaxation to stationary states is constrained by the macroscopic number of conserved quantities afforded by integrability.

Fluctuations in non-equilibrium transport can be studied by analyzing the statistics of number of particles their energy or any charge they carry, passing through an interface in a bi-partition of the system. In an ensemble formulation, fluctuations originate from the initial state.

Our analytical approach to solving the problem is via GHD and Non-linear fluctuating hydrodynamics. We now turn our attention to the well-known partitioning protocol, where two homogeneous, semi-infinite systems are stitched together at the space-time point (x=50, t=0). The two sub-systems have different initial root densities causing a flow of charges between the two sub-systems once they are joined together. The temperature T = 293.15, however it would also be interesting to see what would happen if each subsystem has a different temperature. In fig () we have plotted the quantity showing the propagation of density-density, momenta-momenta and energy-energy correlations.

We now move on to the actual numerical results, for the time-integrated fields and currents we see a smooth decline in the density of the particles, this means that averaged over all the time steps the density remains lower in the right half of the volume compared to the left half. What is equally interesting is the momenta, we see a pyramid shape emerge for the time-integrated current averaged over many initial realizations, this is because in the inhomogeneous situation, quantities change in time, it is no longer stationary and hence the time average is no longer equivalent to the sample average. Momenta increases as particles start moving from left to right, and momenta are higher in the middle as these start moving first.

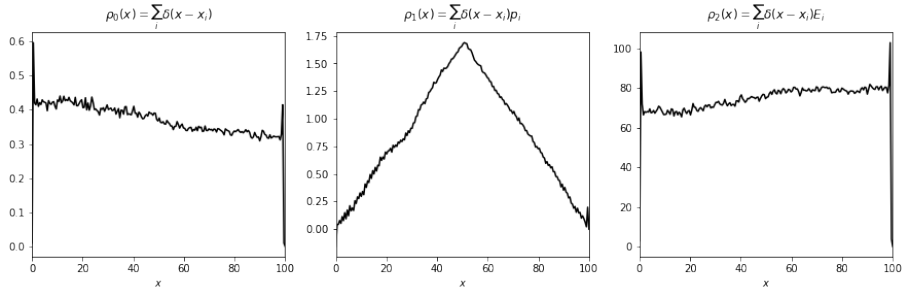

Figure 8: These are time integrated charges for density, momenta and energy for a system of hard rods with equal masses prepared in an initial domain wall state. We can see that there is smooth transition from the high density region to the low density region. We also note that for a state spaces the density of particles remains higher in the left as compared to the right. We expect this graph to smooth out fully and become constant if we were to run the simulation for much larger time scales.

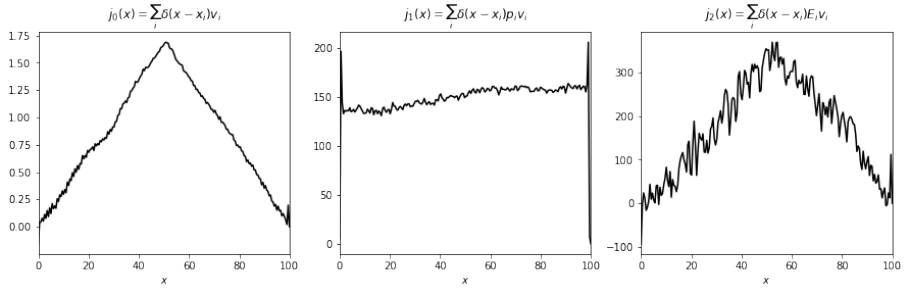

Figure 9: These are the time integrated currents for the hard rods with equal unit masses. The current density has the same structure as the charge momenta. The momenta current remains more or less constant. The energy current also has a structure similar to the current density albeit a more noisy version, which can be explained due to the small sample size.

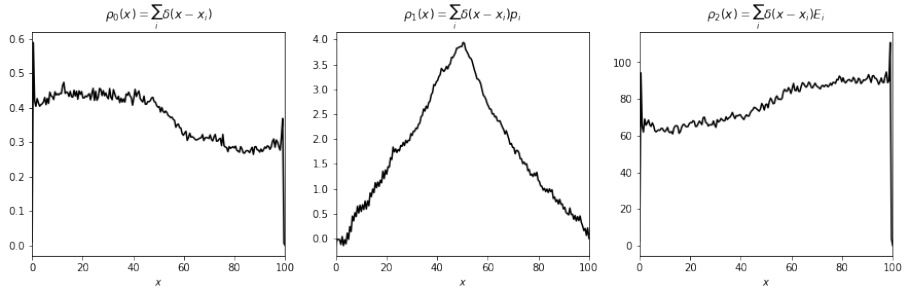

Figure 10: These are time-integrated charges for the density, momenta and energy for hard rods of random masses with an initial domain wall condition. We can see that the density has a much less smoother transition from the high density to low density regions. The structure of the time integrated momenta remains the same as the equal mass, the only different is the size of the peak, where the charge reaches its maximum at approximately 4

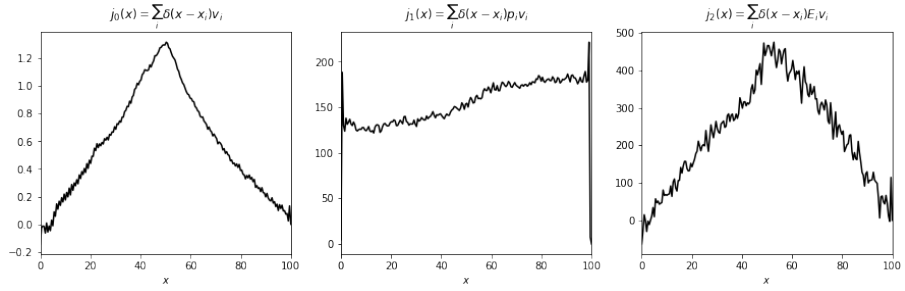

Figure 11: These are time-integrated currents for the density, momenta and energy for hard rods of random masses with an initial domain wall condition. The structure of the current are similar to that of the equal mass hard rods.

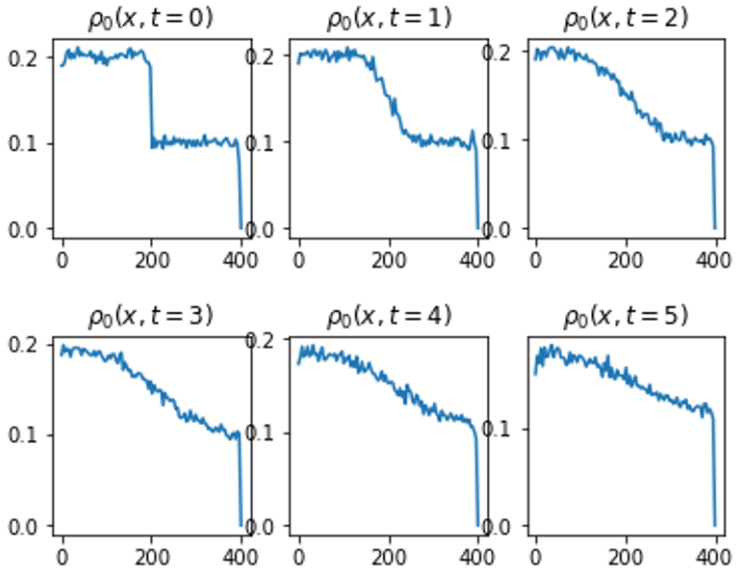

Figure 12: These are the density functions plotted over space for specific snapshots in time. We see that there is steep drop in the density for t=0, this is a shock wave that is formed by mixing two systems that are not in the same equilibrium. The model used over here is the equal mass hard rods. Due to integrability we see a much smoother transition from the high density to low density region.

The theoretical solution to the Riemann Problem for hard rods with equal masses is solved here [4]. We seek to replicate the results as well as apply the simulation to hard rods with random masses to compare how the densities reach an equilibrium state.

The Rod length a=1, initial condition on the state:

$$\rho(x, t=0) = \begin{cases} 0.2, & \text{if } x_i \in [0, L/2] \\ 0.1, & \text{if } x_i \in (L/2, L] \end{cases} \tag{2.30}$$

Where L = 400, The temperature $T = 293.15$ and the number of samples averaged over is 500

These are an interesting set of graphs. We can see clearly that there is a difference between how the densities transition from a high-energy region to a low energy region for the integrable and non-integrable systems.

In figure 12, the graphs smooth out very quickly and looks close to a straight line while figure 13 still has a sharp jump. In fact, if we increase the number of particles we expect the sharp jump in figure 13 to be more prominent, and these sharp jumps are interpreted as shock waves, albeit very weak shock waves. The reason we see longer shock waves in figure 13 compared to figure 12 is because

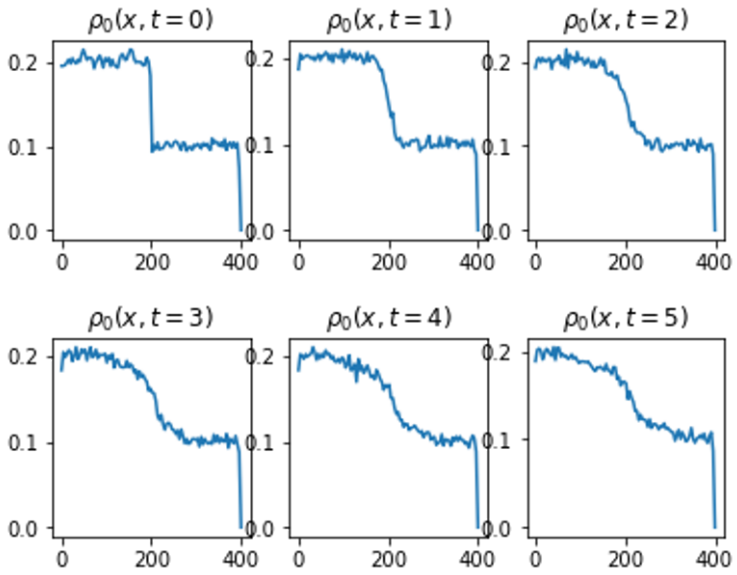

Figure 13: Density for a mixture of hard rods with unequal masses and a non-equilibrium initial state at times t=0,1,2,3,4,5

of the diffusive effects that smooth the graph out, where the ballistic effects are much weaker in the integrable model. The integrable system has a lot of conserved quantities such as the velocities, thus there are a lot of modes that move ballistically (like a tagged particle trajectory) these modes carry different quantities which spreads the observables. In the non-integrable system, there are only a limited amount of conserved quantities, which in turn results in far fewer modes that can carry different quantities, hence the observables do not spread that much. There is some intial smoothing because of diffusion, however this is a very slow effect and will only have a dominant effect during large time scales. Thus, the expected theory is confirmed by the graphs.

We can also numerically calculate the shock curves by finding the non-trivial solutions to the Rankine-Hugoniot jump condition [12]

$$(v - v_0) = j(\vec{u}) - j(\vec{u_0}) \tag{2.31}$$

However this a different problem and will not be tackled in this project

3 Dynamical Correlation Functions

In this section we numerically calculate the dynamical Euler-scale space-time correlation function by numerically solving equations. This procedure is based on linear responses and an extension of the fluctuation dissipation principle. First, we calculate the spreading of correlations from an initial homogeneous thermal state, then we move on to the Riemann problem. As a classical model, we can simulate this using Monte-Carlo and molecular dynamics simulations the details of which are given in the next section. Once we get the trajectories, we can calculate the transport coefficients and measure the microscopic spreading of correlations between them. By doing this we can get a rough estimate on how much time it takes for the correlations to decay as well as compare the accuracy of our code to previous results.Next, we identify the propagator, propagating from time t = 0 to time t as a linear response to the variations on the initial conditions.

The spreading of correlations in a homogeneous state is simple as the diffusion D = 0 causing the propagator to vanish. Further the velocity of the quasi particles is spatially independent. The correlations actually spread at the same velocity of the quasi particles while they decay over time at the rate 1/t. This 1/t decay is a special property found in integrable systems caused by the continuum of hydrodynamic modes.

The space time correlations are dominated by their direct contributions arising from the propagation of quasi particles. Therefore, the spreading of correlations within a given model is highly dependent on its underlying dispersion relation. This is illustrated perhaps most clearly by comparing the spreading of correlations in a relativistic and in a non-relativistic model prepared in similar initial states. It is important to note that Euler hydrodynamics does not fully determine the correlation functions, and although Generalized Gibbs Ensembles provide all the information needed to determine the full thermodynamics of the system, it cannot sufficiently determine correlations of type $< q_i(x)q_j(0) >$.

The correlations can be seen as being produced by waves of conserved quantities ballistically propagating in the fluid between the fields involved in the correlation function. The main idea is to use responses to local disturbances cantered at the volume in order to generate dynamical correlations. Equilibrium time correlations help explain the behaviour of hard rod fluids for various initial conditions near thermal equilibrium. There are many observables we can correlate but the most interesting would be the conserved fields. The diffraction patterns obtained as a Fourier transformation in space and time of these correlations are seen in neutron scattering experiments.

It assumed that for large time and space scales, the time correlations for these conserved fields are governed by a linearized version of its hydro-dynamical equations. Thus the hard rods with varying initial conditions are a simple

example to test and confirm these theories.

The field-field correlator is defined as

$$S_{(m,n)}(x,t) =< Q_m(x,t)Q_n(x,0) > - < Q_m(x,t) >< Q_n(x,0) > \tag{3.1}$$

where $< . >$ is the sample average and $m, n = 1, 2, 3$. We can also extract from the field-field correlator the static co-variance matrix C as

$$S_{(m,n)}(x,0) = C_{m,n}\delta(x) \tag{3.2}$$

similarly we can obtain the current-current correlations and field-current correlations as

The current-current correlator is defined as

$$\prod_{(m,n)}(x,t) =< J_m(x,t)J_n(x,0) > - < J_m(x,t) >< J_n(x,0) > \tag{3.3}$$

where $m, n = 1, 2, 3$ and the field-current correlator is

$$U_{(m,n)}(x,t) =< Q_m(x,t)J_n(x,0) > - < Q_m(x,t) >< J_n(x,0) > \tag{3.4}$$

We have to be more precise when defining our correlation functions, for our simulations x does not refer to a point in space rather a fluid cell of a particular length. Thus we are measuring the densities of of fluid cells and correlating with the fluid cell at the centre of the lattice.

We can rewrite (3.1) and (3.3) as shown in [15]

$$S_{(i,i}(x,t) = \frac{1}{M}\sum_{i=1}^{M}\frac{\rho_i(x,t)\rho_i(L/2,0)}{l} - (\frac{1}{M}\sum_{i=1}^{M}\frac{\rho_i(x,t)}{l})(\frac{1}{M}\sum_{i=1}^{M}\frac{\rho_i(L/2,0)}{l}) \tag{3.5}$$

where M is the number of samples, l is the length of the fluid cell, and L/2 is the fluid cell located at the centre of the lattice. These correlation functions are calculated for $i = 1, 2, 3$

$$\prod_{(i,i}(x,t) = \frac{1}{M}\sum_{i=1}^{M}\frac{J_i(x,t)J_i(L/2,0)}{l} - (\frac{1}{M}\sum_{i=1}^{M}\frac{J_i(x,t)}{l})(\frac{1}{M}\sum_{i=1}^{M}\frac{J_i(L/2,0)}{l}) \tag{3.6}$$

where M is the number of samples, l is the length of the fluid cell, and L/2 is the fluid cell located at the centre of the lattice.

3.1 Field-Field Correlations

The spreading of the correlations in space directly corresponds to the spreading of information on local excitations. For our simulations we consider the expansion dynamics of densities, momentum and energy initially localized at the central region of the 1D lattice.

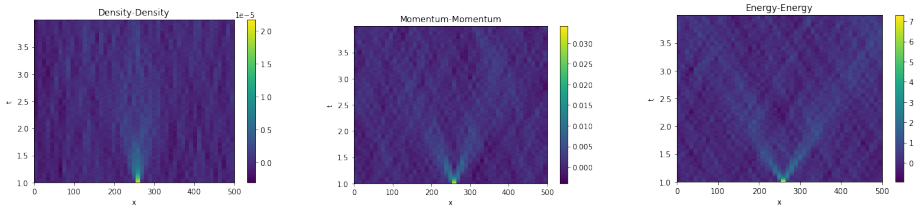

Figure 14: Homogeneous initial condition equal mass

For the correlations of hard rods with equal masses and an equilibrium initial state, we can see that the density-density has a weak linear cone for the zero velocity. What is distinct in the density-density correlations is the shape of the light cone, there seems to be a faint continuum of propagating modes rather than a finite amount, we may have gotten clearer results for larger sample sizes. When running trial simulations, we have observed that the correlations do not have have any interesting structure for sample sizes less than a 100. Thus running simulations of samples with a size of order 10^5 or higher will yield more defined results. Compared to the density-density correlations the momentum-momentum and energy-energy correlations are much more graphic. On average, the two point correlations have a pronounced period of 2. The decay rate i.e. the time taken for the correlations to reach the boundary, are comparatively fast as expected for an integrable system. The momentum-momentum and density-density correlations decay in approximately 4 seconds. The number of particles fluctuate between 230-270 and we observe on average 20,112 collisions during the time evolution of each initial state.

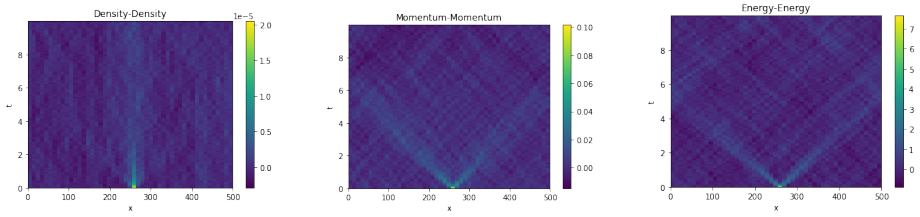

Figure 15: homogeneous initial condition alternating mass correlations

For the hard rods with alternating masses, we can see a a weak linear cone emerging from the zero velocity, supported by light correlations on either sides (illustrated by the contour lines) These 3 modes relate to the 3 conserved quanti-

ties of the system. While it is not as clearly defined as we expected, we can make the calculation of the correlations a bit more accurate by measuring the correlations after the system has been thermalized. The momentum-momentum and energy-energy correlations have a period of 2. The decay rate is slower than that of the equal mass hard rods. The momentum-momentum correlations reach the boundary at t=6, while the energy-energy correlations reach the boundary at approximately t=5.On average, we observe on average 17,996 collisions during the time evolution of each initial state.

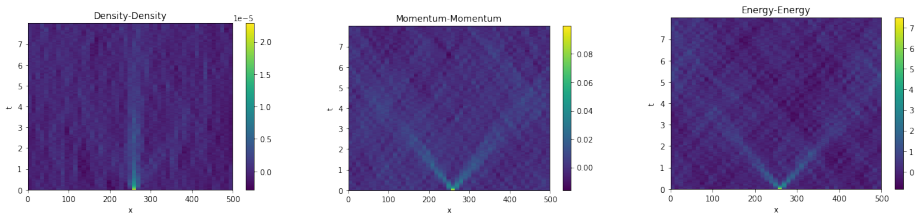

Figure 16: homogeneous initial condition random mass correlations

For the hard rods with unequal masses, as with the alternating mass hard rods, we see a strong correlation at the central lattice site supported by weaker correlations on either side of the linear cone at the zero velocity (illustrated by the contour lines. The momentum-momentum and energy-energy correlations have the same structure as the previous ones with a period of 2. The decay rates are even slower than the alternating mass model; the momentum-momentum correlations reach the boundary at approximately t=8, while the energy-energy correlations reach the boundary at approximately t=6. We observe a slightly less rate of collision compared to the equal mass hard rods, observing on average 15,016 collisions for the time evolution of each initial state.

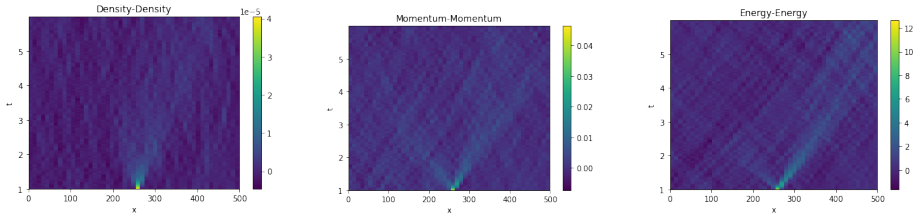

Figure 17: inhomogeneous initial condition equal mass correlations

Now we begin calculating the correlations for hards with an initial domain wall. The density for the first half of the lattice is approximately 0.5 while the other half of the lattice is prepared with a density of 0.15, where we have tried to split the ratio of densities as 3:1. The density-density correlations look interesting and also very different from the homogeneous intial state, mainly because we can see that there is a stronger correlation on the right propagating mode.

The momenta-momenta and energy-energy correlations also show a pattern of stronger correlation on the right propagating mode.

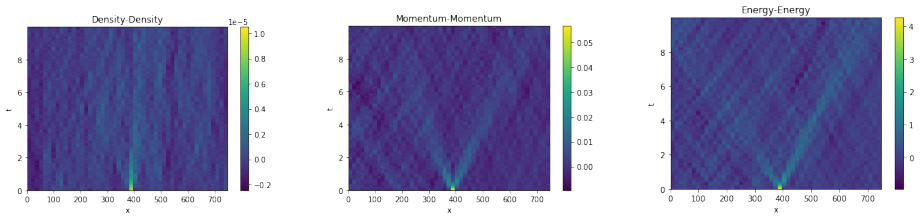

Figure 18: inhomogeneous initial condition alternating mass correlations

The correlations for the alternating mass model when placed in a non-equilibrium starting state is very different compared to the same model with an equilibrium starting state. The density-density graph does not reveal much but we can still see that there is stronger correlations in the central region of the lattice. It is interesting to see that the momentum-momentum takes a lot more time to decay as compared to the 6 seconds taken for the same model with an equilibrium starting state. The correlations for the momentum-momentum reach the boundary at approximately t=12 while the modes for the energy-energy correlations reach the boundary at approximately t=11. As with the density-density correlations we can see a stronger correlation on the right side of the ensemble as opposed to the left. The number of collisions observed on average throughout the simulation is approximately 12,882 which is less than the average collision rate for the equal mass hard rods prepared in a non-equilibrium starting state.

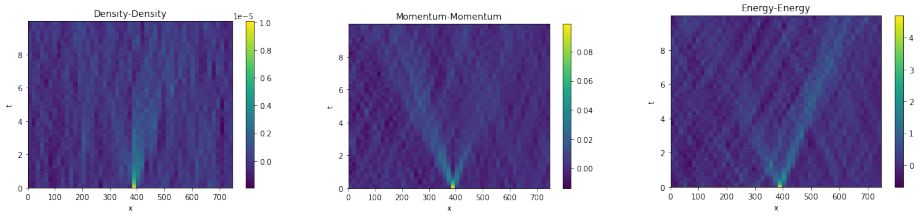

Figure 19: inhomogeneous initial condition random mass correlations

The correlations for the unequal mass hard rods with an domain wall imposed on its initial state also give an interesting pattern. The correlations structures are similar with the density-density having a period of 1 while the momenta-momenta has a period of 2. One can also notice slightly stronger correlations on the right side of the density-density correlations and energy-energy correlations compared to stronger correlations on the left side of the momenta-momenta.. This asymmetry reflects the net flow of quasi particles from right to left. The momentum-momentum and the energy-energy correlations have 2

modes each. With the left mode being more visible for the former and the opposite for the latter. It is interesting to note that when the particles are placed in a non-equilibrium starting state, the emergence of shocks and rarefaction waves also hold important information about the transport of the system.

3.2 Current-Current Correlations

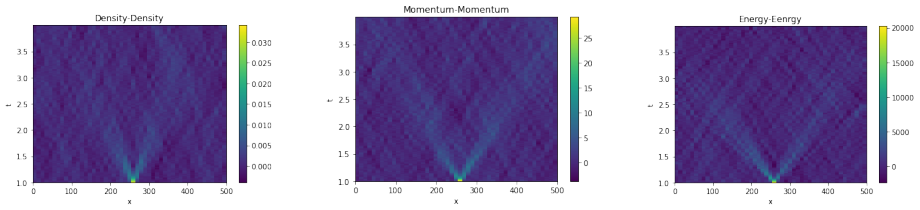

Figure 20: homogeneous initial condition equal mass particle correlation functions

The current-current correlations are much more frustrating to analyze, seeing how 1/3 of the simulation results came up inconclusive. For the hard rods with equal mass and homogeneous initial conditions, the current-current correlations decay quite rapidly compared to the field-field correlations. In-fact the correlations reach the boundary at approximately half the time as compared the field-field correlation i.e t=2. Another interesting thing to note is the frequency of the modes. We observed a linear cone at the zero velocity for the density-density correlations supported by weaker correlations on either side (field-field version), however for the density-density correlations (current-current version) we see 2 modes of non-zero velocities.

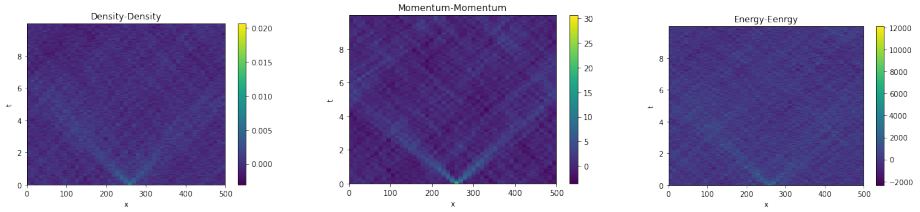

Figure 21: homogeneous initial condition alternating mass particle correlations

The correlations for the alternating mass model decay slower than than the equal mass correlations. The correlations reach the boundary at approximately t=5. This is approximately the same as the field-field correlations. As with the previous current-current correlations we observe a period of 2. The energy-energy correlations are a lot noisier compared to density-density and momentum-momentum correlations. Thus it is very hard to extract any meaningful information from it.

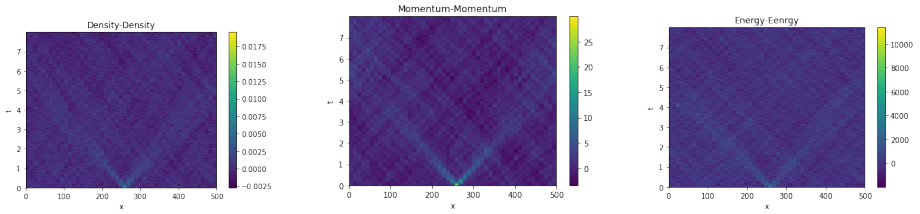

Figure 22: homogeneous initial condition random mass particle correlations

Now we move on to the random mass current-current correlations. We see weak propagating modes of period 2 for the density-density, the correlations decay at approximately t=5. The momenta-momenta correlations are a little bit more defined compared to the density and momenta. The modes seem to reach the boundary at approximately the same time, i.e t=5.

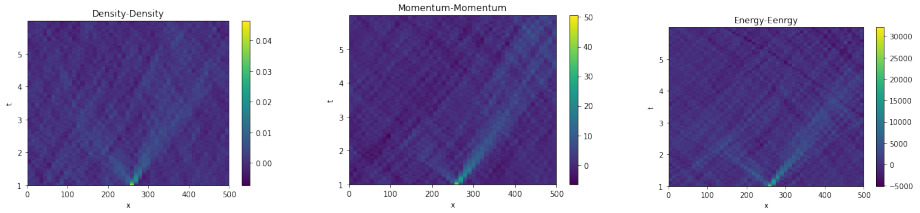

Figure 23: inhomogeneous initial condition equal mass particle correlations

The inhomogeneous current-current correlations see to have a similar structure to the field-field correlators. Mainly we notice that one of the modes are more strongly correlated than the other. However we do not see an anti-symetric pattern across all three graphs like we did for fig 17

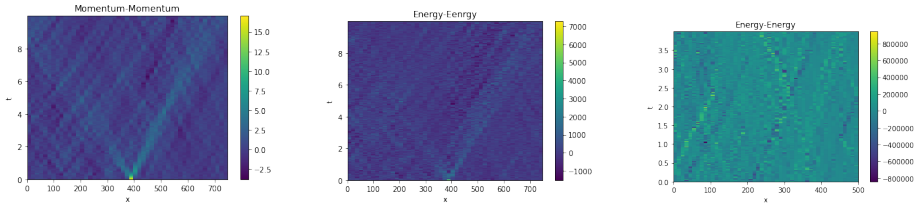

Figure 24: inhomogeneous initial condition alternating mass particle correlations

The alternating mass model seems to have very little correlations for the energy-energy and while we can see a very weak pattern emerge for the momenta-momenta, we cannot approximate the decay rate. The density-density correlations have a stronger correlation on the right mode as compared to the left one.

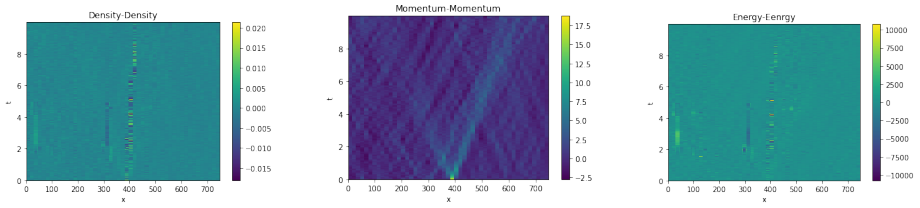

Figure 25: inhomogeneous initial condition random mass particle correlations

Unfortunately, the results for the random mass particles did not show any meaningful results, there are zero correlations for the density-density and the energy-energy. We suspect that the correlations are very hard to extract for these 2 cases and may have required a more precise definition or maybe a higher number of initial samples to average over. The momenta-momenta correlations do look well defined with a similar pattern as to other in-homogeneous correlation functions as the right mode seems to have higher correlations than the left one.

4 Computational Fluid Dynamics

Computational Fluid Dynamics is a powerful tool to analyse systems of fluid flow, heat transfer and other phenomena arising due to chemical reactions by using computers to simulate physical models. It spans a wide range of applications aerodynamics, hydrodynamics, chemical engineering and oceanography, the list is in-exhaustive. The first recorded history of using Monte Carlo simulations was in the 1930's by Enrico Fermi while studying neutron diffusion. In the late 1940's, nuclear physicists working on the Manhattan project developed the modern version of Markov Chain Monte Carlo. The Monte-Carlo technique was a useful tool to study thermal properties of system by the means of random sampling, however there was one drawback, the actual time dynamics of the system would be lost. To get around this physicists relied on numerically solving newton's equations for many body problems.

In the late 1950's, Alder and Wainwright used an IBM 704 computer to simulate perfectly elastic collisions between hard spheres. In the 1960's, as more interest developed in the field of CFD and computer power improved, we began to achieve realistic results for radiation damage of copper, simulations of liquid argon, and self diffusion coefficients all of which compared well with experimental data.

CFD can be thought off as a group of computational mathematical techniques implemented in a particular sequence to solve equations governing fluid flow. We make certain physical assumptions as well as constraints on the system including boundary conditions, ensemble volume, temperature, etc. One of the fundamental ideas governing CFD is the existence of conservation laws, namely conservation of mass, linear momentum and energy.

Once the initial configuration is obtained, the simulation is started and the equations are solved iteratively. The particle trajectories are stored in an 2 dimensional matrix and the observables are calculated in a fluid cell approximation. A fluid cell is on mesoscopic scale somewhere between the microscopic and macroscopic dynamics.

4.1 Details of Monte Carlo Simulations and Molecular Dynamics

We present here additional details about the Monte-Carlo simulations. In our simulations we fix the initial point L=0 where rods are distributed in space, however we keep the number of particles N fluctuating around the required density. While it is more accurate to keep N fixed, we do not expect to see any drastic change in the actual numerical results. The position of the initial rod keeps fluctuating upon every realization of the hard rods configuration and to it we add the minimum distance needed between any two rods (this is 1 since each of our rods are of unit length, the distance between their centers is also 1).

The simulations are performed in finite volume, with the signs of the velocities being reversed with every boundary collision. We only have to worry about the first and last particles interacting with the boundary as their order remains

the same throughout the simulation. The velocities are drawn from a Maxwell Boltzmann distribution which is just a gaussian distribution with $\mu = 0$ and $\sigma^2 = \sqrt{\frac{k_B \times T}{m}}$ where k_B is the Boltzmann constant and T is temperature set at 293.15 unless stated otherwise.

The densities, momentum, energy and correlations have been calculated by averaging over the number of Monte-Carlo sampled initial conditions.

Where n(x, t) and n(0,0) denote the number of rods at time t and an interval [x-l/100, x+l/100] and at time t =0 and in [245-255]. We choose a density of 0.5 for the number of particles, the prediction interval for the number of hard rods is [230-270].

The first step is to import all the necessary modules.

```
import numpy as np
import matplotlib.pyplot as plt
import itertools as it
from matplotlib.animation import FuncAnimation
from numba import jit
from time import time
from numpy.random import random
from scipy.fft import fft2
import os
```

Listing 1: Python Code

When sampling positions for each particle we need to ensure that no two particles overlap. Since our code records the mid point of the hard rod, we can ensure this condition is met by looking at the absolute difference in distances between particles i and j. If we do indeed find an overlap between any two particles, we can simply disqualify this configuration and run the code until the next valid configuration is determined. We can speed this process by sorting the positions in ascending order and checking the distance between only the neighbouring particles i and i+1.

```
def is_legal(x, delta):
    for pair in it.permutations(x,2):
        if abs(pair[0]-pair[1]) < delta:
            return False
    return True

def sample_positions(N, L, delta, n):

    configs = np.zeros((n,N))
    i = 0

    while i != n:
        x = (L - delta)* np.random.random(N) + delta/2
        if is_legal(x, delta):
            configs[i] = x
            i = i+1

    return configs
```

Listing 2: Python Code

We can run this snipped to visualize what 20 different Monte Carlo configurations for our ensemble.

```
L = 20
N = 20
delta = 0.4
n = 10

configs_10  = sample_positions(L, N, delta, n)
```

Listing 3: Python Code

```
fig, ax = plt.subplots(dpi=100)

for i in range(n):
    ax.axhline(i, lw=0.5, c="k", ls=":")
    ax.errorbar(configs_10[i], [i]*N, xerr = delta/2, ls="", lw=2,
    capsize=2, capthick=2, c="k")

ax.set(xlim=(0,L), xlabel = "$x$", ylabel = "configuration", yticks
     = range(n));
```

Listing 4: Python Code

The next step is to generate velocities for the hard particles. We chose the velocities from a Maxwell-Boltzmann Distribution. This is simply a gaussian function with $\mu = 0$ and $\sigma^2 = \sqrt{\frac{k_B T}{m}}$

```
def sample_velocities(N, m, T, n):

    k_b = 1.38064
    sigma = np.sqrt((k_b * T)/m)
    v = np.random.normal(loc=0, scale=sigma, size = (n, N))
    return v
```

Listing 5: Python Code

```
N = 5
n = 1000
m = np.random.random((1,N))*10
T = 300

v = sample_velocities(N, m, T, n)
```

Listing 6: Python Code

When running simulations of the order of 1000 or higher, especially with time-evolution, it is essential to reduce the code complexity but at the same time creating the right conditions for the system. We can modify our original MC code by using the following iterative method.

We place the 1st particle at x_0. To get the position of the n^{th} particle we add the minimum distance needed between the n^{th} and the $(n+1)^{th}$ particles which in this case is just the length of one particle "a". To this we add a random positive number using the Poisson function. One of the disadvantages would be the fact that the number of particles inside the finite volume keep fluctuating

but for large systems the overall density deviates very little from the average number of particles that can be placed in the ensemble. However we find that the complexity of this system becomes $O(n)$.

```
class Monte_Carlo_H:
    def __init__(self, L, delta = 1, T=1):
        self.L = L # length of simulation box
        self.delta = delta # size of the particles
        self.T = T # temperature

    def sample_configuration(self):
        x = self.sample_positions()
        N = len(x)
        m = np.random.random(N)*10
        v = self.sample_velocities(N, m)
        print(x)
        print(N)
        return x, v, m

    def sample_velocities(self, N, m):
        k_b = 1.38064
        sigma = np.sqrt((k_b * self.T)/m)
        v = np.random.normal(loc = 0, scale = sigma, size=N)
        return v

    def sample_positions(self):
        x_positions = []
        next_position = 0.5 + np.random.poisson(1)
        while next_position < self.L - 0.5:
            x_positions.append(next_position)
            next_position = x_positions[-1] + 1 + np.random.poisson
    (1)
        return x_positions
```

Listing 7: Python Code

For the Monte Carlo method with equal unit masses for each particle we simply need to change our definition for m to create an array of length N consisting of ones. This can be done with built in functions available in python.

```

class Monte_Carlo:
    def __init__(self, L, delta = 1, T=1):
        self.L = L # length of simulation box
        self.delta = delta # size of the particles
        self.T = T # temperature

    def sample_configuration(self):
        x = self.sample_positions()
        N = len(x)
        m = np.ones(N)
        v = self.sample_velocities(N, m)
        print(x)
        print(N)
        return x, v, m

    def sample_velocities(self, N, m):
        k_b = 1.38064
```

```
        sigma = np.sqrt((k_b * self.T)/m)
        v = np.random.normal(loc = 0, scale = sigma, size=N)
        return v

    def sample_positions(self):
        x_positions = []
        next_position = 0.5 + np.random.poisson(2)
        while next_position < self.L - 0.5:
            x_positions.append(next_position)
            next_position = x_positions[-1] + 1 + np.random.poisson
    (1)
        return x_positions
```

Listing 8: Python Code

Coding the alternating mass problem requires a little more involvement. Since the number of particles in each Monte Carlo sample keeps fluctuating we need the masses to stop when the number of particles in that particular sample is done. We can achieve this by assigning a RV1 to even numbers and another RV2 to odd numbers. This will help us understand conserved quantities and correlations in bi-atomic systems. We can choose a mass ratio greater than or equal to 3. This will ensure sufficient amounts of chaos in the system. We have taken our mass ratio to be 5.

```

class Monte_Carlo:
    def __init__(self, L, delta = 1, T=1):
        self.L = L # length of simulation box
        self.delta = delta # size of the particles
        self.T = T # temperature

    def sample_configuration(self):
        x,m = self.sample_positions()
        N = len(x)
        v = self.sample_velocities(N, m)
        print(m)
        return x, v, m

    def sample_velocities(self, N, m):
        k_b = 1.38064
        sigma = np.sqrt((k_b * self.T)/m)
        v = np.random.normal(loc = 0, scale = sigma, size=N)
        return v

    def sample_positions(self):
        x_positions = []
        m = np.array([])
        next_position = 0.5 + np.random.poisson(2)
        i = 0
        m1 = 1
        m2 = 5
        while next_position < self.L - 0.5:
            if i%2 == 0:
                m = np.append(m,m1)
            else:
                m = np.append(m,m2)
            if next_position < self.L - 0.5:
```

```
                x_positions.append(next_position)
                next_position = x_positions[-1] + 1 + np.random.
    poisson(1)
            i = i+1
        return x_positions, m
```

Listing 9: Python Code

We now code the case for the Riemann Problem. Our goal is to change the densities of the system for two equal halves of the ensemble. We can modify our "sample position" class to account for the change in density by passing two different parameters through the Poisson function. In our case we take the mean $\mu = 2$ for the volume from [0,L/2] and a mean $\mu = 4$ for the volume [L/2, L]. On average we expect 125 particles to be arranged in the ensemble. We can confirm this by plotting the densities for time t=0. There is sharp jump in the density at L/2.

```

class Monte_Carlo:
    def __init__(self, L, delta = 1, T=293.15):
        self.L = L # length of simulation box
        self.delta = delta # size of the particles
        self.T = T # temperature

    def sample_configuration(self):
        x = self.sample_positions()
        N = len(x)
        m = np.random.random(N)*10
        v = self.sample_velocities(N, m)
        print(x)
        print(N)
        return x, v, m

    def sample_velocities(self, N, m):
        k_b = 1.38064
        sigma = np.sqrt((k_b * self.T)/m)
        v = np.random.normal(loc = 0, scale = sigma, size=N)
        return v

    def sample_positions(self):
        x_positions = []
        next_position = 0.5 + np.random.poisson(1)
        while next_position < self.L - 0.5:
            if next_position < self.L/2:
                x_positions.append(next_position)
                next_position = x_positions[-1] + 1 + np.random.
    poisson(2)
            else:
                x_positions.append(next_position)
                next_position = x_positions[-1] + 1 + np.random.
    poisson(4)
        return x_positions
```

Listing 10: Python Code

```

```

```
def sample_configuration(self):
        x = self.sample_positions()
        N = len(x)
        m = np.ones(N)
        v = self.sample_velocities(N, m)
        print(x)
        print(N)
        return x, v, m
```

Listing 11: Python Code

As done for the homogeneous case, we can also implement the alternating mass and equal unit mass cases by looping over every odd and even index.

```

class Monte_Carlo:
    def __init__(self, L, delta = 1, T=1):
        self.L = L # length of simulation box
        self.delta = delta # size of the particles
        self.T = T # temperature

    def sample_configuration(self):
        x,m = self.sample_positions()
        N = len(x)
        v = self.sample_velocities(N, m)
        print(m)
        return x, v, m

    def sample_velocities(self, N, m):
        k_b = 1.38064
        sigma = np.sqrt((k_b * self.T)/m)
        v = np.random.normal(loc = 0, scale = sigma, size=N)
        return v

    def sample_positions(self):
        x_positions = []
        m = np.array([])
        next_position = 0.5 + np.random.poisson(2)
        i = 0
        m1 = 1
        m2 = 5
        while next_position < self.L - 0.5:
            if i%2 == 0:
                m = np.append(m,m1)
            else:
                m = np.append(m,m2)
            if next_position < self.L/2 - 0.5:
                x_positions.append(next_position)
                next_position = x_positions[-1] + 1 + np.random.
    poisson(2)
            else:
                x_positions.append(next_position)
                next_position = x_positions[-1] + 1 + np.random.
    poisson(4)
            i = i+1
        return x_positions, m
```

Listing 12: Python Code

We now turn to the actual time evolution of the system. This is captured by molecular dynamics simulations. We create various classes such as "simulation" which integrates newton's equations of motion using a simple 1st order Runge Kutta method, i.e Euler's method. We can also use other techniques of integration such as Verlet's algorithm to increase the precision and decrease the time complexity. The most important events during the Molecular Dynamics simulations is the collision between any two particles as well as collision with the boundary walls. We have implemented the algorithm discussed in section 3.2 to calculate the velocities of the i and j particles after collision. The velocities for the $0^{th} and n^{th}$ particles switch between a negative and positive sign every time the particles reflect from the boundary walls.

```
1
2  class Molecular_Dynamics:
3      def __init__(self, N, L, dt, t_max, x, v, m, delta=1):
4          self.L = L
5          self.t_max = t_max
6          self.dt = dt
7          self.m = m
8          self.delta = delta
9          self.x = x
10         self.v = v
11         self.x_data = [x]
12         self.v_data = [v]
13         self.t = [0]
14         self.pair_idx = tuple((i, i+1) for i in range(N-1))   #
       pair index
15         self.single_idx = (0, N-1)
16         self.number_of_events = 0
17
18
19     def simulation(self):
20         while self.t[-1] < t_max:
21             t_event, event_idx = self.compute_next_event()
22             for i in range(int(self.t[-1]/self.dt)+1, int((self.t
       [-1]+t_event)/dt)+1):
23                 self.x = np.copy(self.x + self.v * self.dt)
24                 self.x_data.append(self.x)
25                 self.v_data.append(self.v)
26                 self.t.append(self.t[-1] + self.dt)
27                 if self.t[-1] >= self.t_max: break
28             self.update_v(event_idx)
29             self.number_of_events += 1
30
31     def get_data(self, p):
32         x_data = np.array(self.x_data)[:-1]
33         v_data = np.array(self.v_data)[:-1]
34         E_data = 0.5 * self.m * v_data**2
35         t = np.array(self.t)[:-1]
36         if p: print("number of collisions", self.number_of_events)
37         return x_data, v_data, E_data, t
38
39     def pair_time(self, xi, xj, vi, vj):
40         dv = vi - vj
41         if dv < 0: return np.inf
```

```
        return (xj - xi - self.delta)/dv

    def pair_collision(self, vi, vj, mi, mj):
        vi_f = vi*(mi-mj)/(mi+mj) + vj*2*mj/(mi+mj)
        vj_f = vi*2*mi/(mi+mj) - vj*(mi-mj)/(mi+mj)
        return vi_f, vj_f

    def wall_time(self, x0, xN, v0, vN):
        t0 = (x0 - self.delta/2)/abs(v0) if v0 < 0 else np.inf
        tn = (L - xN - self.delta/2)/vN if vN > 0 else np.inf
        return [t0, tn]

    def wall_collision(self, vi):
        return -vi

    def compute_next_event(self):
        x, v = self.x, self.v
        pair_times = [self.pair_time(x[i], x[j], v[i], v[j]) for i,
     j in self.pair_idx]
        wall_times = self.wall_time(x[0], x[-1], v[0], v[-1])
        times = np.array(wall_times + pair_times)
        arg = np.argmin(times)
        t_event = times[arg]
        event_idx = self.single_idx[arg] if arg < 2 else self.
    pair_idx[arg-2]
        return t_event, event_idx

    def update_v(self, event_idx):
        v = self.v.copy()
        if type(event_idx) == tuple:
            i, j = event_idx # i and the i+1 element pair(i,j)
            v[i], v[j] = self.pair_collision(v[i], v[j], m[i], m[j
    ])
        else:
            i = event_idx
            v[i] = self.wall_collision(v[i])
        self.v = v
```

Listing 13: Python Code

The observables of interest in this system are the conserved fields and their Euler currents. These are the density fields, momentum fields and energy fields. We can calculate the density as per the provided in section 3.3. We start by creating an equally spaced grid with n number of bins, say L/a. We then check if a particle particle lies in a specific dx. If the function returns the value true, we assign the normal integer value and sum over all particles for that particular point in space. We repeat this process for every time step, adding the instantaneous values to an empty matrix of size (L/a, t/dx). This is matrix with values 1 wherever a particle is found and 0 otherwise. We can normalize this function dividing each of the 1's by the total number of particles. Thus if we sum over all the values in observable(x,t) for any specific t, we get the total sum = 1. We can similarly compute the momentum fields and energy fields by multiplying this matrix with the momentum and energy of the particle respectively. This 2 dimensional matrix will be used to calculate the hydrodynamic matrices and

correlation functions.

```
1
2  def observables(nbins, x_data, v_data, E_data, m, N):
3      x_grid = np.linspace(0, L, nbins)
4      dx = x_grid[1]
5
6      enumerate(x_grid)
7
8      density0 = np.zeros((len(t),len(x_grid)))
9      density1 = np.zeros((len(t),len(x_grid)))
10     density2 = np.zeros((len(t),len(x_grid)))
11
12     j0 = np.zeros((len(t),len(x_grid)))
13     j1 = np.zeros((len(t),len(x_grid)))
14     j2 = np.zeros((len(t),len(x_grid)))
15
16     for ((x_, v_), (E_, m_)) in zip(zip(x_data.T, v_data.T), zip(
       E_data.T, m)):
17         for i, a in enumerate(x_grid):
18             idx = np.logical_and(a < x_, x_ < a + dx)
19             density0[:,i] += idx.astype(int)
20             density1[:,i] += v_*m_*idx.astype(int)
21             density2[:,i] += E_*idx.astype(int)
22             j0[:,i] += v_*idx.astype(int)
23             j1[:,i] += v_*m_*v_*idx.astype(int)
24             j2[:,i] += E_*v_*idx.astype(int)
25
26
27     density0 = density0 / N
28     density1 = density1 / N
29     density2 = density2 / N
30
31     j0 = j0 / N
32     j1 = j1 / N
33     j2 = j2 / N
34
35     return x_grid, density0, density1, density2, j0, j1, j2
```

Listing 14: Python Code

This is the main code used to iterate between the Monte Carlo samples and the time evolution of each initial distribution. We fix the length i.e volume of the ensemble in such a way that the particle density is approximately 0.5. The volume for the hard rods without an initial domain wall is 500 and the volume with an initial domain wall is 750. The time we run the code for is non-trivial, we first calculate the correlations for a reasonable time-scale and then adjust the time to avoid the effects of boundary collisions in further simulations. This block of code is especially long as it contains the correlation functions needed to calculate the static co-variance matrix and the Density-Current correlations. However due to computational and time constraints we are unable to verify any of the matrices. In addition, we measure our original correlations from time t=0, however we include calculations of correlations from time t=0.2, t=0.5 and t=1.0. The reason behind this being that the system will be in a thermalized state at later times as compared to time t=0. This may improve the results for

our correlations. Unfortunately, we were unable to test this theory out, with trial calculations coming inconclusive.

```
L = 500
t_max = 8
dt = 0.001

MC = Monte_Carlo(L, T = 293.15)

n_MC = 1000

for i in range(n_MC):
    x, v, m = MC.sample_configuration()

    N = len(x)

    MD = Molecular_Dynamics(N, L, dt, t_max, x, v, m)
    MD.simulation()

    x_data, v_data, E_data, t = MD.get_data(p = True)

    nbins = 50

    x_grid, q0, q1, q2, j0, j1, j2 = observables(nbins, x_data,
    v_data, E_data, m, N)

    mid = q0.shape[1]//2 #Half of the box (position we want to
    correlate)#

    maxtime = 8000

    if i == 0:
        term11 = q0[0:maxtime, :] * q0[0,mid]
        term21 = q0[0:maxtime,:]
        term31 = q0[0,mid]

        term12 = q1[0:maxtime, :] * q1[0,mid]
        term22 = q1[0:maxtime,:]
        term32 = q1[0,mid]


        term13 = q2[0:maxtime, :] * q2[0,mid]
        term23 = q2[0:maxtime,:]
        term33 = q2[0,mid]


        term14 = j0[0:maxtime, :] * j0[0,mid]
        term24 = j0[0:maxtime,:]
        term34 = j0[0,mid]


        term15 = j1[0:maxtime, :] * j1[0,mid]
        term25 = j1[0:maxtime,:]
        term35 = j1[0,mid]


        term16 = j2[0:maxtime, :] * j2[0,mid]
        term26 = j2[0:maxtime,:]
```

```
        term36 = j2[0,mid]

        print(i)

    else:
        term11 += q0[0:maxtime, :] * q0[0,mid]
        term21 += q0[0:maxtime,:]
        term31 += q0[0,mid]

        term12 += q1[0:maxtime, :] * q1[0,mid]
        term22 += q1[0:maxtime,:]
        term32 += q1[0,mid]

        term13 += q2[0:maxtime, :] * q2[0,mid]
        term23 += q2[0:maxtime,:]
        term33 += q2[0,mid]

        term14 += j0[0:maxtime, :] * j0[0,mid]
        term24 += j0[0:maxtime,:]
        term34 += j0[0,mid]

        term15 += j1[0:maxtime, :] * j1[0,mid]
        term25 += j1[0:maxtime,:]
        term35 += j1[0,mid]

        term16 += j2[0:maxtime, :] * j2[0,mid]
        term26 += j2[0:maxtime,:]
        term36 += j2[0,mid]


        print(i)

C0 = (term11 / n_MC) - ((term31 / n_MC)*(term21 / n_MC))
C1 = (term12 / n_MC) - ((term32 / n_MC)*(term22 / n_MC))
C2 = (term13 / n_MC) - ((term33 / n_MC)*(term23 / n_MC))
C3 = (term14 / n_MC) - ((term34 / n_MC)*(term24 / n_MC))
C4 = (term15 / n_MC) - ((term35 / n_MC)*(term25 / n_MC))
C5 = (term16 / n_MC) - ((term36 / n_MC)*(term26 / n_MC))
```

Listing 15: Python Code

```

fig, (ax1, ax2) = plt.subplots(1,2)


T, X = np.meshgrid(t[:maxtime], x_grid)
C0FT = fft2(C0)

pcm1 = ax1.pcolormesh(X, T, C0.T)


pcm2 = ax2.pcolormesh(X, T,np.abs(C0FT).T, vmin = 0, vmax = 20)

fig.colorbar(pcm1, ax = ax1)
fig.colorbar(pcm2, ax = ax2)

```

```
plt.show()
```

Listing 16: Python Code

```

E = np.sum(0.5 * m * v_data**2, axis = 1)

plt.plot(t, E)
plt.xlabel = "time"
plt.ylabel = "Energy"


plt.plot(t, E_data)
plt.xlabel = "time"
plt.ylabel = "Energy"
```

Listing 17: Python Code

We can use this snippet of code to visualize our densities as time averaged values. We can see that the graph looks as expected for the homogeneous and domain wall initial conditions.

```

def observables_TA(nbins, x_data, v_data, E_data, m):
    x_grid = np.linspace(0, L, nbins)
    dx = x_grid[1]

    enumerate(x_grid)

    density0 = np.zeros_like(x_grid)
    density1 = np.zeros_like(x_grid)
    density2 = np.zeros_like(x_grid)

    j0 = np.zeros_like(x_grid)
    j1 = np.zeros_like(x_grid)
    j2 = np.zeros_like(x_grid)

    for (x_, v_), (E_, m_) in zip(zip(x_data.T, v_data.T), zip(
    E_data.T, m)):
        for i, a in enumerate(x_grid):
            idx = np.logical_and(a < x_, x_ < a + dx)
            density0[i] += idx.sum()
            density1[i] += np.sum(v_[idx]*m_)
            density2[i] += np.sum(E_[idx])
            j0[i] += np.sum(v_[idx])
            j1[i] += np.sum(v_[idx]*m_*v_[idx])
            j2[i] += np.sum(E_[idx]*v_[idx])

    density0 = density0 / np.sum(dx*density0)
    density1 = density1 / dx
    density2 = density2 / dx

    j0 = j0 / dx
    j1 = j1 / dx
    j2 = j2 / dx

    return x_grid, density0, density1, density2, j0, j1, j2
```

Listing 18: Python Code

```

nbins = 250
x_grid, density0, density1, density2, j0, j1, j2 = observables_TA(
    nbins, x_data, v_data, E_data, m)
```

Listing 19: Python Code

```

fig, ax = plt.subplots(1, 3, figsize=(15,4))

ax[0].set(xlabel = "$x$", xlim = (0,L), title=r'$\rho_0(x) = \sum_i
     \delta(x-x_i)$')
for i, x_ in enumerate(x_data.T):
    values, _ = np.histogram(x_, bins=x_grid, density=True)
    values = values / np.sum(x_grid[1]*values) / N
    ax[0].plot(x_grid[:-1], values, label='%.1e'%m[i])

ax[0].plot(x_grid, density0, c="k")
ax[0].legend(loc=(-0.5, 0), title = "mass")

ax[1].plot(x_grid, density1, c="k")
ax[1].set(xlabel = "$x$", xlim=(0,L), title=r'$\rho_1(x) = \sum_i \
    delta(x-x_i)p_i$')

ax[2].plot(x_grid, density2, c="k")
ax[2].set(xlabel = "$x$", xlim=(0,L), title=r'$\rho_2(x) = \sum_i \
    delta(x-x_i)E_i$')
```

Listing 20: Python Code

```
#Density at specific time
fig_den= plt.figure()
ax1 = fig_den.add_subplot(231)
ax1.plot(x_grid, density0[0,:])
ax1.set_title(r'$\rho$_0(x,0)')

#Density averaged over time
ax2 = fig_den.add_subplot(232)
ax2.plot(x_grid, np.mean(density0, axis = 0))
ax2.set_title('Mean density')

plt.plot(x_grid, C0[0])
plt.plot(x_grid, C0[1000])
plt.plot(x_grid, C0[2000])
plt.plot(x_grid, C0[3000])
plt.plot(x_grid, C0[4000])
plt.plot(x_grid, C0[5000])
plt.show()

plt.plot(x_grid, C0[0])
plt.plot(x_grid, C0[2000])
plt.plot(x_grid, C0[4000])
plt.plot(x_grid, C0[5999])
plt.show()

plt.plot(x_grid, C0[0])
plt.show()
```

```
28
29 plt.plot(x_grid, C0[1000])
30 plt.show()
31
32 plt.plot(x_grid, C0[2000])
33 plt.show()
34
35 plt.plot(x_grid, C0[3000])
36 plt.show()
37
38 plt.plot(x_grid, C0[4000])
39 plt.show()
40
41 plt.plot(x_grid, C0[5000])
42 plt.show()
```

Listing 21: Python Code

```
1 print('B00 is', np.sum(C00[0]))
2 print('B01 is', np.sum(C01[0]))
3 print('B02 is', np.sum(C02[0]))
4 print('B10 is', np.sum(C10[0]))
5 print('B11 is', np.sum(C11[0]))
6 print('B12 is', np.sum(C12[0]))
7 print('B20 is', np.sum(C20[0]))
8 print('B21 is', np.sum(C21[0]))
9 print('B22 is', np.sum(C22[0]))
```

```
1 plt.plot(x_grid, C00[0], label = 't=0')
2 plt.plot(x_grid, C00[1000], label = 't=1')
3 plt.plot(x_grid, C00[2000], label = 't=2')
4 plt.plot(x_grid, C00[3000], label = 't=3')
5 plt.plot(x_grid, C00[4000], label = 't=4')
6 plt.plot(x_grid, C00[5000], label = 't=5')
7 plt.xlabel('x')
8 plt.ylabel('C00')
9 plt.legend()
10 plt.show()
11
12 plt.plot(x_grid, C00[0], label = 't=0')
13 plt.plot(x_grid, C00[2000], label='t=2')
14 plt.plot(x_grid, C00[4000], label='t=4')
15 plt.plot(x_grid, C00[5999], label='t=6')
16 plt.xlabel('x')
17 plt.ylabel('C00')
18 plt.legend()
19 plt.show()
20
21 plt.plot(x_grid, C00[0])
22 plt.show()
23
24 plt.plot(x_grid, C00[1000])
25 plt.show()
26
27 plt.plot(x_grid, C00[2000])
28 plt.show()
29
```

```
plt.plot(x_grid, C00[3000])
plt.show()

plt.plot(x_grid, C00[4000])
plt.show()

plt.plot(x_grid, C00[5000])
plt.show()
```

5 Conclusion

To conclude this project, we give a brief summary of the results as well as talk about what would have been interesting to see if we had a larger time-frame. The space time correlation functions which decay anomalously have faster decay rates for integrable systems compared to non-integrable systems. Further when an initial domain wall is imposed the correlations decay even slower than when it is in equilibrium initial states. The densities charges tend to reach an equilibrium in a smoother manner for the integrable case as opposed to the non-integrable case.

The hydrodynamic matrices that were introduced in sections GHD and NLFH are of particular importance. This is because the eigenvalues of the static co-variance matrix gives us the velocities of the propagators in the correlation functions. We tried to prove this however it required considerable efforts on the coding. Of course, this is very non-trivial and has not actually been verified numerically before thus it would be a nice project idea.

Another concept that could have probably given us better simulation data as well as more accurate dynamical structure factors is the use of quantum Monte-Carlo methods instead of the classical Monte-Carlo algorithms used throughout this project. Of course, as the name suggests, this would mean solving the Schrodinger equation on a quantum computer, which would be difficult to obtain as not many exist in the public domain.

As with any simulation, we find that the results drastically change with the number of samples averaged over, we were limited to a 1000 to get results in a day. However we also included some code that allowed us to save just the final correlation functions, allowing us to run numerous simulations over an extended period of time, say 30 days, giving us approximately 30,000 samples, however we did not have time to implement and test these changes, thus we only include it in section 5 as a suggestion.

Another interesting area to investigate would be the shock waves and rarefaction patterns. These are obtained by solving the Cauchy problem.

Overall, when I approached professor Benjamin for a computationally rigorous project, I may have hit a little bit above my weight, nevertheless over the course of this project I have not only learned a lot about hydrodynamics but also a lot about coding, especially how complicated debugging code can be.

During the first few months my molecular dynamics algorithms neither allowed for interaction with particles nor did they remain in the box. while the results of this project are definitely non-trivial there is still a lot more areas to improve and build further on and I hope to work on such computational projects in the future as well.

A Dynamical Structure Factor

In this section we obtain the dynamical structure factor for a random collection of correlation functions. The dynamical structure was obtained by taking the Fourier transformation of every column converting the space-time domain to the space-frequency domain, then we take the inverse Fourier transformation along all the rows of the space-frequency domain to get the wave number-frequency domain. Finally we plot the absolute values as heat maps. These dynamical structure factors are directly seen in neutron scattering experiments [17], however there isn't enough literature for me to make a conclusion thus I include it as only an appendix.

Since a physical interpretation of applying the Fourier transformation to correlation functions would seem similar to taking an x-ray of a fluid, however we are working in one spatial dimension making all of this a naive speculation.

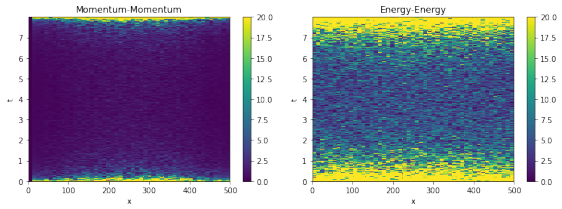

Figure 26: Fourier transformation of momenta current-current correlations and energy current-current correlations for hard rods with alternating masses in an equilibrium initial state.

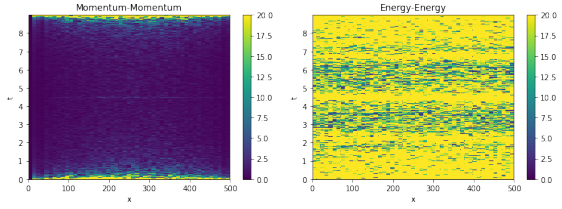

Figure 27: Fourier transformation of momenta current-current correlations and energy current-current correlations for hard rods with random masses in an equilibrium initial state.

References

[1] C Boldrighini, RL Dobrushin, and Yu M Sukhov. "One-dimensional hard rod caricature of hydrodynamics". In: *Journal of Statistical Physics* 31.3 (1983), pp. 577–616.

[2] Olalla A. Castro-Alvaredo, Benjamin Doyon, and Takato Yoshimura. "Emergent Hydrodynamics in Integrable Quantum Systems Out of Equilibrium". In: *Physical Review X* 6.4 (Dec. 2016). ISSN: 2160-3308. DOI: 10.1103/physrevx.6.041065. URL: http://dx.doi.org/10.1103/PhysRevX.6.041065.

[3] Benjamin Doyon. "Lecture notes on Generalised Hydrodynamics". In: *SciPost Physics Lecture Notes* (Aug. 2020). ISSN: 2590-1990. DOI: 10.21468/scipostphyslectnotes.18. URL: http://dx.doi.org/10.21468/SciPostPhysLectNotes.18.

[4] Benjamin Doyon and Herbert Spohn. "Dynamics of hard rods with initial domain wall state". In: *Journal of Statistical Mechanics: Theory and Experiment* 2017.7 (July 2017), p. 073210. ISSN: 1742-5468. DOI: 10.1088/1742-5468/aa7abf. URL: http://dx.doi.org/10.1088/1742-5468/aa7abf.

[5] Leonhard Euler. "Principes généraux du mouvement des fluides". In: *Mémoires de l'Académie des Sciences de Berlin* (1757), pp. 274–315.

[6] Gabriele Giuliani and Giovanni Vignale. *Quantum theory of the electron liquid*. Cambridge university press, 2005.

[7] Pierre C Hohenberg. "Existence of long-range order in one and two dimensions". In: *Physical Review* 158.2 (1967), p. 383.

[8] DW Jepsen. "Dynamics of a simple many-body system of hard rods". In: *Journal of Mathematical Physics* 6.3 (1965), pp. 405–413.

[9] V Kozlov. "Integrability and non-integrability in Hamiltonian mechanics". In: *Russian Mathematical Surveys* 38.1 (1983), pp. 1–76.

[10] Aritra Kundu and Abhishek Dhar. "Equilibrium dynamical correlations in the Toda chain and other integrable models". In: *Physical Review E* 94.6 (Dec. 2016). ISSN: 2470-0053. DOI: 10.1103/physreve.94.062130. URL: http://dx.doi.org/10.1103/PhysRevE.94.062130.

[11] Olga Aleksandrovna Ladyzhenskaya. *The mathematical theory of viscous incompressible flow*. Vol. 2. Gordon and Breach New York, 1969.

[12] Christian B Mendl and Herbert Spohn. "Shocks, rarefaction waves, and current fluctuations for anharmonic chains". In: *Journal of Statistical Physics* 166.3-4 (2017), pp. 841–875.

[13] Christian B. Mendl and Herbert Spohn. "Equilibrium time-correlation functions for one-dimensional hard-point systems". In: *Physical Review E* 90.1 (July 2014). ISSN: 1550-2376. DOI: 10.1103/physreve.90.012147. URL: http://dx.doi.org/10.1103/PhysRevE.90.012147.

[14] N David Mermin and Herbert Wagner. "Absence of ferromagnetism or antiferromagnetism in one-or two-dimensional isotropic Heisenberg models". In: *Physical Review Letters* 17.22 (1966), p. 1133.

[15] Frederik Skovbo Møller et al. "Euler-scale dynamical correlations in integrable systems with fluid motion". In: *SciPost Physics Core* 3.2 (Dec. 2020). ISSN: 2666-9366. DOI: 10.21468/scipostphyscore.3.2.016. URL: http://dx.doi.org/10.21468/SciPostPhysCore.3.2.016.

[16] Jason Myers et al. "Transport fluctuations in integrable models out of equilibrium". In: *SciPost Physics* 8.1 (Jan. 2020). ISSN: 2542-4653. DOI: 10.21468/scipostphys.8.1.007. URL: http://dx.doi.org/10.21468/SciPostPhys.8.1.007.

[17] Herbert Spohn. "Hydrodynamical theory for equilibrium time correlation functions of hard rods". In: *Annals of Physics* 141.2 (1982), pp. 353–364.

[18] Herbert Spohn. "Interacting and noninteracting integrable systems". In: *Journal of Mathematical Physics* 59.9 (Sept. 2018), p. 091402. ISSN: 1089-7658. DOI: 10.1063/1.5018624. URL: http://dx.doi.org/10.1063/1.5018624.

[19] Herbert Spohn. "Nonlinear Fluctuating Hydrodynamics for Anharmonic Chains". In: *Journal of Statistical Physics* 154.5 (Feb. 2014), pp. 1191–1227. ISSN: 1572-9613. DOI: 10.1007/s10955-014-0933-y. URL: http://dx.doi.org/10.1007/s10955-014-0933-y.